付 1. 分散並列型動的ソルバー主サブルーチン

付 1.1 マスター主サブルーチン

ここでは、マスター側とスレーブ側、双方の動的ソルバーの主サブルーチン Ver.1.11 の全てを示す。解析の流れは、単一 CPU の動的ソルバーとほぼ同一であり、各サブルーチンの動作などの説明は、マニュアル「動的解析編」を参照されたい。マスターとスレーブ相互におけるデータの送受信、スレーブの解析制御をマスター側からどうのように行っているなどを、検証されると良い。

```
C―――――――――――――――――――――――――――――――――――――――――――――――
C―――――――――――――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /submain_dynamic_a
C―――――――――――――――――――――
C         Parallel version for Master
C       ―――――――――――――――――――――
C     ●   動的解析マスター用主プログラム（反復解法）Ver.1.11
C―――――――――――――――――――――――――――――――――――――――――――――――
C―――――――――――――――――――――――――――――――――――――――――――――――
c       マスター並列処理用　メインプログラム
c         マスターとモニターの ID は     0
c         スレーブは                     1    以降連続番号で定義
c       基本計画
c           部材に関する動的領域はマスターは全部材、スレーブは担当部材のみ確保する。
c           要素及び節点データに関しては全てマスターと同じとする
c           自由度に関するデータはマスターは全ての領域、
c           マスターは担当部材が必要とする領域のみ使用する
c           スレーブが必要とする情報は、マスターより転送する
c
c  マスター側がスレーブに送受信するデータブロックは以下のようである。
c     Ⅰ。予備計算
c           1. コントロール情報送信（全スレーブ同じ情報）
c           2. 構造データ送信（全スレーブ同じ情報）
c           3. 初期不整データ送信（全スレーブ同じ情報）注１
c           4. ファイバーデータ送信（全スレーブ同じ情報）注１
c           5. RO モデル用データ送信（全スレーブ同じ情報）注１
c           6. 減衰用データ送信（全スレーブ同じ情報）注１
c           7. 初期変位送信（全スレーブ同じ情報：現在使用不可）
c           8. 初期応力送信（全スレーブ同じ情報：現在使用不可）
c              注１スレーブ側が必要としなくても、全スレーブに転送する。
c     Ⅱ。動的解析
c           1. 右辺項を受信（スレーブによって受信数が異なる）
c           2. 計算結果加速度データ送信（スレーブによって送信数が異なる）
c           3. 未収束コード送信（全スレーブ同じ情報）
c           4. 収束コード送信（全スレーブ同じ情報）
c           5. 部材両端の節点力を受信（スレーブによって受信数が異なる）
c     Ⅲ。後処理
c           1. 最大応力等をスレーブからの受信（スレーブによって受信数が異なる）
c
c  スレーブで送受信数が異なるデータはスレーブ転送用テーブルを用いて、
c  送信する場合は圧縮し、受信する場合は展開する。
c           1. スレーブ転送用テーブル作成   Convert_node_inf_vpp( ) Create_table( )
c           2. 送信圧縮   Dt_from_M_to_S( )
c           3. 受信展開   Dt_from_S_to_M( )
c
c            スレーブ制御コード 0:  解析開始　正常終了の場合は最後は制御コードなし
c                               1:  発散・途中終了
c                               2:  エラーは発生、途中終了
c                               3:  未収束
c
c
c―――――――――――――――――――――――――――――――――――――――――――――――
c       ** 解析番号と担当部材を GUI モニターから取得
cc
c         Model_No.1 = 1              !  幾何学非線形型有限要素弾性モデル
c         Model_No.2 = 2              !  3 次元せん断弾塑性モデル
c         Model_No.3 = 3              !  3 次元軸力弾塑性モデル
c         Model_No.4 = 4              !  3 次元ケーブル弾塑性モデル
c         Model_No.5 = 5              !  3 次元免振モデル（MSS モデル）
c         Model_No.6 = 6              !  3 次元制震 Maxwell モデル
c         Model_No.7 = 7              !  3 次元弾塑性バネモデル(*)
c
c
c         Model_No.11 = 11            !  両端ファイバーモデル
c         Model_No.12 = 12            !  両端、中央ファイバーモデル
c         Model_No.13 = 18            !  両端ピンで中央ファイバーモデル
c         Model_No.15 = 15            !  ファイバーモデル
c
c         Model_No.13 = 13            !  両端 MS モデル
c         Model_No.14 = 14            !  両端、中央 MS モデル
c
c         Model_No.31 = 16            !  両端アナロジーモデル
c         Model_No.32 = 17            !  両端、中央アナロジーモデル
c         Model_No.32 = 19            !  両端ピン、中央アナロジーモデル
c
c         Model_No.51 = 51            !  3 次元プレテンション動作モデル
c
c         Model_No.50 = 1001          !  DLL 有限要素弾塑性モデル
c
c―――――――――――――――――――――――――――――――――――――――――――――――
c
c       ファイバー履歴タイプ
c       nm_type: 1      バイリニア型
c         No.1 = 1                    !  対称バイリニア
```

```
c        No.2 = 2                    !  対称トリリニア
c        No.3 = 3                    !  直線コンクリート型
c        No.4 = 4                    !  曲線線コンクリート型
c        No.5 = 5                    !  対称バイリニア型(移動＋等方効果用)
c        No.6 = 6                    !  対称トリリニア型（移動＋等方効果用）
c        No.7 = 7                    !  非対称バイリニア型
c        No.8 = 8                    !  非対称トリリニア型（移動＋等方効果用）
c
c        アナロジーモデル履歴タイプ
c              11      完全弾塑性型
c              12      等方硬化弾塑性型
c              13      移動硬化弾塑性型
c              14      等方硬化＋移動硬化弾塑性型
c
c
c        マルチスプリング履歴タイプ
c              21      武田モデル
c              22      等方硬化＋移動硬化トリリニア型
c
c
c-------------------------------------------------------------------
c
c      断面モデル
c        No.1                      !  円管
c        No.2                      !  角パイプ
c        No.3                      !  H  型
c        No.4                      !  長方形
c        No.5                      !  T  型
c        No.6                      !
c        No.7                      !
c
c-------------------------------------------------------------------
c
c     部材モデルの階層構造は、以下のようである。
c     部材ー＞要素モデルー＞断面モデルー＞履歴モデル
c     １．部材は、各部材固有のデータを保持する。
c     ２．要素モデルは、部材の内部を構成するデータを保持する。
c     ３．断面モデルは、部材の断面形状を等のデータを保持する。
c     ４．履歴モデルは、各要素の弾塑性状態を制御するデータを保持する。
c
c
c
c     部材モデルの組み込み手順
c     １．Member_s と Element_s 構造体を設定する。
c     ２．弾塑性解析を実行するために必要となる領域（構造体）を設定する。
c     ３．Cal_stiff_linear に線形剛性を組み込む
c     ４．Cal_stress に応力計算を組み込む
c     ５．Check_stress に弾塑性チェックを組み込む
c     ６．Get_nonlinear_stiff に非線形剛性を組み込む
c
c
c
c
c      解析種別  N_analysis :
c          7: 線形解析
c          8: 幾何学的非線形解析
c          9: 弾塑性解析
c         10: 幾何学的＋弾塑性解析
c          6: 線形座屈解析
c
c     部材別解析種別：Member(i).nm_analysis
c         -1: 線形解析
c          0: 全体解析種別に従う
c          1: 部材座標系 x 方向弾性
c          2: 部材座標系 y 方向弾性
c          3: 部材座標系 z 方向弾性
c
c-------------------------------------------------------------------
c
c submain_dynamic_a  引数
c               iend_code       :計算終了コード 0=継続  1:終了
c               icontrol        :計算コード     0:予備計算  1:動的解析 99:終了処理
c               ierr_dat        :エラーコード
c               T               :計算時刻
c               dt              :解析増分時間
c               n_step          :今回の解析ステップ数
c               ns_step         :解析開始ステップ（最初はゼロセットが必要）
c               n_iterate       :反復回数
c-------------------------------------------------------------------
c-------------------------------------------------------------------
c
c     ★         サブルーチン定義
c
c-------------------------------------------------------------------
c-------------------------------------------------------------------
      subroutine submain_dynamic_a(i_calnum,iend_code,icontrol,ierr_dat,
     *            T,dt,n_step,ns_step,d_max_v,id_max_v,
     *            i_read_disp,F_disp,i_read_ndbalanceF,F_ndbalanceF,
     *            i_read_spring,F_fay,F_n_spring,F_my_spring,
     *            F_mz_spring,i_stat_spring,n_iterate,
     *            nm_iterate,numb_method)

c-------------------------------------------------------------------
c-----------------------------------------------★並列処理に必要な宣言
      use MPI_DEFINE
      use TIME_MODULE                          ! 時間計測用定義・変数
c-------------------------------------------------------------------
      implicit real*8(A-H,O-Z)
```

```
c————————————————————————————————★動的解析制限条件
      parameter(lAnalysis_number_condition = 0)
      parameter(Number_of_nodes    = 250)
      parameter(Number_of_members = 500)
c————————————————————————————————★計算結果のダンプ出力ファイル番号
      parameter(damp_out = 76)
c————————————————————————————————★構造体の定義ヘッダーファイル
      include "..¥..¥sf3st¥submain.h"
      include "..¥..¥sf3st¥submainx.h"
      include "..¥..¥sf3st¥New_submain.h"
c————————————————————————————————————————————————————————
c
c     ★          静的配列定義
c
c————————————————————————————————————————————————————————
      dimension ifl(16),ifly(16),idfile(100),iflz(16)
      character fnx_file*100,TITLEX*50
      dimension fs_st(10,3),fl_st(10,3),ifp_st(10,3),ist(3)
      dimension igra(3),xgal(3),hh(100),qhh(100),vacc(6)
      dimension No_section(10)
      dimension M_alloc(10)                        ! 動的配列確保のチェック
      dimension dt_M_filter(4)                     ! Maxwell   フィルターデータ
      real*8 nStart                                ! 時間計測用
      integer    S_control_code                    ! スレーブコントロールコード
c————————————————————————————————————————————————————————
c
c     ★     コントローラ GUI からのインターフェイスデータ
c
c————————————————————————————————————————————————————————
      real*4  F_disp(3,*)
      real*4  F_ndbalanceF(3,*),
     *        F_fay(5,*),
     *        F_n_spring(5,*),
     *        F_my_spring(5,*),
     *        F_mz_spring(5,*)
      integer i_stat_spring(5,*)
c————————————————————————————————————————————————————————
c
c     ★     解析手法のコントロールデータ
c
c————————————————————————————————————————————————————————
      integer N_implicit_method     !陰解法の回数   －１：無限回
      integer Iteration_method      !反復計算：１、(陰解法：２:使用せず)
c————————————————————————————————————————————————————————
c
c     ★          動的配列定義
c
c————————————————————————————————————————————————————————
c      O）データ取得バッファ
      integer, save, ALLOCATABLE  :: nm_parallel(:,:)
      real*8,  save, ALLOCATABLE  :: pa_work_force(:,:)
      real*4, ALLOCATABLE :: ff_data(:)
      integer, ALLOCATABLE ::if_data(:),iif_dt(:),ibuf(:)
      real*8, ALLOCATABLE :: buf(:),buf_disp(:)
      integer, ALLOCATABLE ::Ms_table(:,:),Ms_free(:)  !加速度のスレーブ転送用テーブル
c      １） スカイライン行列
      real*8, ALLOCATABLE :: gskym(:),gskym_d(:),twork(:)
      integer,ALLOCATABLE :: nwork(:)
c————————————————————————————————————————————————————————
c       real*8  gskym(n_skyline)        ! スカイライン行列
c————————————————————————————————————————————————————————
c      ２）自由度関連
      real*8, ALLOCATABLE ::
     *   disp_point(:),vel_point(:),acc_point(:),
     *   est_disp_point(:),est_vel_point(:),est_ddisp_point(:)
      real*8, ALLOCATABLE ::
     *   result_disp_point(:),result_vel_point(:),result_acc_point(:),
     *   past_disp_point(:),past_vel_point(:),past_acc_point(:),
     *   past_dacc_point(:)
      real*8, ALLOCATABLE ::
     *   ld_point(:),ld_point_repeat(:),fld_static(:,:),
     *   a_vector(:),b_vector(:)
      integer, ALLOCATABLE :: max_h_sky(:)
c————————————————————————————————————————————————————————
c       real*8  disp_point(n_unknown)    ! 節点変位
c       real*8  vel_point(n_unknown)     ! 節点速度
c       real*8  acc_point(n_unknown)     ! 節点加速度

c       real*8  est_disp_point(n_unknown)    ! 予測節点変位
c       real*8  est_vel_point(n_unknown)     ! 予測節点速度
c       real*8  est_ddisp_point(n_unknown)   ! 予測節点増分変位

c       real*8  result_disp_point(n_unknown)    ! 計算結果節点変位
c       real*8  result_vel_point(n_unknown)     ! 計算結果節点速度
c       real*8  result_acc_point(n_unknown)     ! 計算結果節点加速度
c       real*8  past_dacc_point(n_unknown)      ! 計算結果節点増分加速度

c       real*8  past_disp_point(n_unknown)    ! Δt 秒前の節点変位
c       real*8  past_vel_point(n_unknown)     ! Δt 秒前の節点速度
c       real*8  past_acc_point(n_unknown)     ! Δt 秒前の節点加速度

c       real*8  ld_point(n_unknown)            ! 線形右辺項ベクトル
c       real*8  ld_point_repeat(n_unknown)     ! 反復用右辺項ベクトル
c
c       real*8  a_vecter(n_unknown)            ! 動的解析用 work ベクトル
c       real*8  b_vector(n_unknown)            ! 動的解析用 work ベクトル
c
c       real*8  fld_static(3,n_unknown)        !  節点荷重の分布（釣合式に用いた座標系）
```

```
c      real*8  fld_dynamic(3,n_unknown)       !  地震に対する荷重ベクトル
c
c      integer  max_h_sky(n_unknown+1)        !  スカイライン行列の各列の高さ
c-------------------------------------------------------------------
c     ３） 地震波の数
      real*8, ALLOCATABLE :: acc_earth(:,:),fdd_point(:,:)
c-------------------------------------------------------------------
c     real*8  acc_earth(3,n_acc_earth)       !  地震波の加速度（時刻歴）
c     real*8  fdd_point(3,n_point)           !  節点荷重（時刻歴）
c-------------------------------------------------------------------
c     ４） 節点
      real*8, ALLOCATABLE :: fll_static_point(:,:,:),
     *                       am_point(:,:),fll_force_point(:,:)
c-------------------------------------------------------------------
c      real*8  fll_static_point(3,6,n_point) !  節点荷重の分布（全体座標系）
c      real*8  fll_force_point(3,n_point)    !  節点不釣合い力の分布（全体座標系）
c      real*8  am_point(2,n_point)           !  節点集中質量
c-------------------------------------------------------------------
c     ５） 部材
      real*8, ALLOCATABLE :: am_member(:,:,:),ac_member(:,:,:)
c-------------------------------------------------------------------
c      real*8  am_member(12,12,n_member)     ! 質量整合行列（釣合座標系）
c      real*8  ac_member(12,12,nc_member)    ! 線形部材減衰行列（釣合座標系）
c-------------------------------------------------------------------
c      ６） 座標変換行列
      real*8, ALLOCATABLE ::rot_memb_t(:,:,:),rot_memb(:,:,:,:),
     *                      rot_local(:,:,:)
c-------------------------------------------------------------------
c      real*8  rot_memb_t(3,3,n_member)        !  部材の全体座標系への回転行列
c      real*8  rot_memb(3,3,2,n_member)        !  部材両端の釣合座標系への回転行列
c      real*8  rot_local(3,3,  n_local_coord)  !  全体座標系から局所座標への回転行列
c-------------------------------------------------------------------
c      ７）剛性行列等
      real*8, ALLOCATABLE ::ak_linear(:,:,:),ak_nonlinear(:,:,:)
c-------------------------------------------------------------------
c       real*8  ak_linear(12,12,n_member)      !  部材の線形剛性行列（釣合座標系）
c       real*8  ak_nonlinear(12,12,n_member)   !  部材の接線剛性行列（部材座標系）
c-------------------------------------------------------------------
c      ８）  部材の非線型状態を表す。ワークエリア
      real*8, ALLOCATABLE ::work1_elememt(:),work2_elememt(:),
     *                      work1_member(:),work2_member(:)
c-------------------------------------------------------------------
c      real*8, ALLOCATABLE ::test_vector(:)
c      real*8   work1_elememt(n_request1* n_element_dll)
c      real*8   work2_elememt(n_request2* n_element_dll)
c      real*8   work1_member(n_request3* n_member_dll)
c      real*8   work2_member(n_request4* n_member_dll)
c      注：上記のワークエリアは DLL 内で structure を定義して使用する。
```



```
c-------------------------------------------------------------------
c
c    ★        構造体定義
c
c-------------------------------------------------------------------
      record /control_s      / Control
      record /dynamic_load_s / Dynamic_load
      record /parameter_s    / Parameter_C
      record /n_model_s      / Model_type
      record /newmark_s      / Newmark_P
      record /out_section_s  / Out_section
      record /point_s        / Point
      record /element_s      / Element
      record /member_s       / Member
      record /max_disp_s     / Max_disp
      record /max_stress_s   / Max_stress
c-------------------------------------------------------------------
c
c    ★        新構造体定義
c
c-------------------------------------------------------------------
      record / S_comp_model_s / S_comp_model
      record / E_modelx_s     / E_modelx
      record / M_modelx_s     / M_modelx
      record / M_model_mss_s  / M_model_mss
      record / M_model_ang_s  / M_model_ang
      record / E_Fiber_Work_s / E_Fiber_Work
      record / M_Fiber_Work_s / M_Fiber_Work
c-------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c
c-------------------------------------★Model_No.2 3 次元せん断弾塑性モデル
      record / RO_work_s /  RO_work
c-------------------------------------★Model_No.3 3 次元軸力弾塑性モデル
c      record /element3_s / Element
c      record /member3_s  / Member
c-------------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
c      record /element4_s / Element
c      record /member4_s  / Member
c-------------------------------------★Model_No.5 3 次元 MSS 免振モデル
      record / MSS_work_s   / MSS_work
c-------------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      record / E_model6_real_s   / E_model6_real
c-------------------------------------★Model_No.7 3 次元バネモデル
c      record /element7_s / Element
c      record /member7_s  / Member
c      record / E_model7_real_s    / E_model7_real
c-------------------------------------★Model_No.11 両端ファイバーモデル
      record / E_model11_s    / E_model11
      record / M_model11_s    / M_model11
```

```
c————————————————————————————★Model_No.12 両端、中央ファイバーモデル
      record / E_model12_s    / E_model12
      record / M_model12_s    / M_model12
c————————————————————————————★Model_No.13 両端ピン、中央ファイバーモデル
      record / E_model13_s    / E_model13
      record / M_model13_s    / M_model13
c————————————————————————————★Model_No.15 両端ファイバーFEM モデル
      record / E_model15_s    / E_model15
      record / M_model15_s    / M_model15
c————————————————————————————★Model_No.21 両端 MS モデル
      record / E_model21_s    / E_model21
      record / M_model21_s    / M_model21
c————————————————————————————★Model_No.22 両端、中央 MS モデル
      record / E_model22_s    / E_model22
      record / M_model22_s    / M_model22
c————————————————————————————★Model_No.31 両端アナロジーモデル
      record / E_model31_s    / E_model31
      record / M_model31_s    / M_model31
c————————————————————————————★Model_No.32 両端、中央アナロジーモデル
      record / E_model32_s    / E_model32
      record / M_model32_s    / M_model32
c————————————————————————————★Model_No.33 両端ピン、中央アナロジーモデル
      record / E_model33_s    / E_model33
      record / M_model33_s    / M_model33
c————————————————————————————★Model_No.51 3 次元プレテンション動作モデル
c      record / E_model51_s    / E_model51
c      record / M_model51_s    / M_model51
c————————————————————————————★ファイバーモデル用エリア
      record / E_model_fiber_s   / E_model_fiber
      record / M_model_fiber_s   / M_model_fiber
c————————————————————————————★履歴モデル用エリア
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
c————————————————————————————————————————————————————————————
c
c    ★          新構造体の動的領域宣言
c
c————————————————————————————————————————————————————————————
      ALLOCATABLE :: S_comp_model(:)
      ALLOCATABLE :: E_modelx(:)
      ALLOCATABLE :: M_modelx(:)
      ALLOCATABLE :: M_model_mss(:)
      ALLOCATABLE :: M_model_ang(:)
      ALLOCATABLE :: E_Fiber_Work(:)
      ALLOCATABLE :: M_Fiber_Work(:)
c————————————————————————————————————————————————————————————
c
c    ★          構造体の動的領域宣言
c
c————————————————————————————————————————————————————————————
      ALLOCATABLE :: Bilinear_work(:)
      ALLOCATABLE :: Trilinear_work(:)
      ALLOCATABLE :: Concrete_work(:)
      ALLOCATABLE :: Point(:)
      ALLOCATABLE :: Element(:)
      ALLOCATABLE :: Member(:)
      ALLOCATABLE :: Max_disp(:)
      ALLOCATABLE :: Max_stress(:)
c————————————————————————————★Model_No.1 通常の有限要素弾塑性モデル
c————————————————————————————★Model_No.2 3 次元せん断弾塑性モデル
      ALLOCATABLE :: RO_work(:)
c————————————————————————————★Model_No.3 3 次元軸力弾塑性モデル
c————————————————————————————★Model_No.4 3 次元ケーブル弾塑性モデル
c————————————————————————————★Model_No.5 3 次元免振モデル
      ALLOCATABLE :: MSS_work(:)
c————————————————————————————★Model_No.6 3 次元制震 Maxwell モデル
      ALLOCATABLE :: E_model6_real(:)
c————————————————————————————★Model_No.7 3 次元バネモデル
c       ALLOCATABLE ::  E_model7(:)
c————————————————————————————★Model_No.11 両端ファイバーモデル
      ALLOCATABLE :: E_model11(:)
      ALLOCATABLE :: M_model11(:)
c————————————————————————————★Model_No.12 両端、中央ファイバーモデル
      ALLOCATABLE :: E_model12(:)
      ALLOCATABLE :: M_model12(:)
c————————————————————————————★Model_No.13 両端ピン、中央ファイバーモデル
      ALLOCATABLE :: E_model13(:)
      ALLOCATABLE :: M_model13(:)
c————————————————————————————★Model_No.15 両端ファイバーFEM モデル
      ALLOCATABLE :: E_model15(:)
      ALLOCATABLE :: M_model15(:)
c————————————————————————————★Model_No.21 両端 MS モデル
      ALLOCATABLE :: E_model21(:)
      ALLOCATABLE :: M_model21(:)
c————————————————————————————★Model_No.22 両端、中央 MS モデル
      ALLOCATABLE :: E_model22(:)
      ALLOCATABLE :: M_model22(:)
c————————————————————————————★Model_No.31 両端アナロジーモデル
      ALLOCATABLE :: E_model31(:)
      ALLOCATABLE :: M_model31(:)
c————————————————————————————★Model_No.32 両端、中央アナロジーモデル
      ALLOCATABLE :: E_model32(:)
      ALLOCATABLE :: M_model32(:)
c————————————————————————————★Model_No.33 ピン 両端、中央アナロジーモデル
      ALLOCATABLE :: E_model33(:)
      ALLOCATABLE :: M_model33(:)
c————————————————————————————★Model_No.51 3 次元プレテンション動作モデル
```

```
c       ALLOCATABLE :: E_model51(:)
c       ALLOCATABLE :: M_model51(:)
c-------------------------------------------------★ファイバーモデル用エリア
      ALLOCATABLE :: E_model_fiber(:)
      ALLOCATABLE :: M_model_fiber(:)
c-----------------------------------------------------------------------
c
c     ★         新構造体の静的確保
c
c-----------------------------------------------------------------------
      save S_comp_model
      save E_modelx
      save M_modelx
      save M_model_mss
      save M_model_ang
      save E_Fiber_Work
      save M_Fiber_Work
c-----------------------------------------------------------------------
c
c     ★         重要データの静的確保
c
c-----------------------------------------------------------------------
      save ifl ,ifly ,idfile ,iflz
      save fnx_file ,TITLEX ,M_alloc
      save Parameter_C,Control,Dynamic_load,Model_type,Newmark_P
      save Point,Element,Member,Max_disp,Max_stress
      save gskym,gskym_d,nwork,twork
      save disp_point,vel_point,acc_point,
     *     est_disp_point,est_vel_point,est_ddisp_point
      save result_disp_point,result_vel_point,result_acc_point,
     *     past_disp_point,past_vel_point,past_acc_point,
     *     past_dacc_point
      save ld_point,ld_point_repeat,fld_static
      save a_vector,b_vector
      save max_h_sky
      save acc_earth
      save fll_static_point,  fdd_point,am_point,fll_force_point
      save am_member,ac_member
      save rot_memb_t,rot_memb,rot_local
      save ak_linear,ak_nonlinear
      save work1_elememt,work2_elememt,work1_member,work2_member

      save Bilinear_work,Trilinear_work,Concrete_work
      save Out_section,n_istep
      save S_control_code
c     save test_vector
c-------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c-------------------------------------★Model_No.2 3次元せん断弾塑性モデル
      save RO_work
c-------------------------------------★Model_No.3 3次元軸力弾塑性モデル
c-------------------------------------★Model_No.4 3次元ケーブル弾塑性モデル
c-------------------------------------★Model_No.5 3次元免振モデル
      save MSS_work
c-------------------------------------★Model_No.6 3次元制震 Maxwell モデル
      save E_model6_real
c-------------------------------------★Model_No.7 3次元バネモデル
c     save E_model7
c-------------------------------------★Model_No.11 両端ファイバーモデル
      save   E_model11
      save   M_model11
c-------------------------------------★Model_No.12 両端、中央ファイバーモデル
      save   E_model12
      save   M_model12
c-------------------------------------★Model_No.13 両端ピン、中央ファイバーモデル
      save   E_model13
      save   M_model13
c-------------------------------------★Model_No.15 両端ファイバーFEM モデル
      save   E_model15
      save   M_model15
c-------------------------------------★Model_No.21 両端 MS モデル
      save   E_model21
      save   M_model21
c-------------------------------------★Model_No.22 両端、中央 MS モデル
      save   E_model22
      save   M_model22
c-------------------------------------★Model_No.31 両端アナロジーモデル
      save   E_model31
      save   M_model31
c-------------------------------------★Model_No.32 両端、中央アナロジーモデル
      save   E_model32
      save   M_model32
c-------------------------------------★Model_No.33 両端ピン、中央アナロジーモデル
      save   E_model33
      save   M_model33
c-------------------------------------★Model_No.51 3次元プレテンション動作モデル
c     save    E_model51
c     save    M_model51
c-------------------------------------★ファイバーモデル用エリア
      save    E_model_fiber
      save    M_model_fiber
c-------------------------------------★解析手法
      save N_implicit_method
      save Iteration_method,stimes
c-------------------------------------★スレーブ転送用テーブル
      save Ms_table,Ms_free,buf_disp
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
```

```
c     ★          システムからの制御情報を取得（以後実行文）
c
c―――――――――――――――――――――――――――――――――――――――――――――――――――――
c―――――――――――――――――――――――――――――――――――――――――――――――――――――
      if(icontrol .eq. 1  ) goto 9990    ! 解析処理へ
      if(icontrol .eq. 99 ) goto 9997    ! 終了処理へ
      if(icontrol .eq. 98 ) goto 9998    ! 途中終了処理へ
      do i=1,10
      M_alloc(i)=0                       ! 動的配列の確保をチェックする配列をゼロセット
      enddo
      open (damp_out,FILE='DOUTPUT')
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★計測用データの初期設定
      All_Time_PRE=0.              ! 予備計算
      All_Time_K=0.                ! 係数行列作成
      All_Time_NEWMARK=0.          ! Newmarkβ法
      All_Time_LD=0.               ! 右辺項作成
      All_Time_NONLINER=0.   ! 非線形項作成
      All_Time_CAL=0.              ! 振動方程式を解く
      All_Time_DISP_VEL=0.   ! 変位、速度を計算
      All_Time_STRESS=0.           ! 応力の計算
      All_Time_F_STRESS=0.   ! ファイバー応力の計算
      All_Time_KT=0.               ! 接線剛性の計算
      All_Time_NET_LD=0.           ! 右辺項の通信
      All_Time_NET_ACC=0.          ! 予測加速度の通信
      All_Time_NET_CONV=0.   ! 収束コードの通信
      All_Time_NET_FORCE=0.  ! 部材節点力の通信
      All_Time_STEP=0.       ! ステップ毎の通信
      All_Time_ALL=0.              ! 総解析時間
c――――― ★★  システムからのコントロール情報を取得
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★MPI 上での自己の状態を得る
      ! 解析プロセスの総数を取得
      call MPI_Comm_size(MPI_COMM_WORLD, n_proc, ierr)
      ! 自分のランクを取得
      call MPI_Comm_rank(MPI_COMM_WORLD, n_myRank, ierr)
      write(damp_out,'(a,2i5)')'(n_proc, myRank) =', n_proc, n_myRank
      ALLOCATE (nm_parallel( 2, 0:n_proc))
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★担当部材リストを取得
      do i=0, n_proc-1
      call RequestMember(nm_parallel, i)
      enddo
      n_member1=nm_parallel(1, n_myRank)       ! 担当部材の先頭番号
      n_member2=nm_parallel(2, n_myRank)       ! 担当部材の最終番号
      max_member=n_member2-n_member1+1       ! 担当部材の最大数
      write(damp_out,'(a,i5,a,i5,a,i5)')'max_member=: cc',max_member,
     +      ' n_member1=',n_member1,' n_member2=',n_member2
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★システムからのコントロール情報を取得(ok)
      call sysnam(FNX_file,N_analysis)
      if(N_analysis.ne.i_calnum) N_analysis=i_calnum  ! 解析番号を変更
      if(N_analysis.le.6) then
      ierr_dat=1
      call err_outf(ierr_dat)
      return
      endif
      ihan = 0
      ierr = 0
      NFILE=100
      ierr_dat =0
      write(damp_out,*) ' System file input ok. No. of analysis:',
     *                  N_analysis
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★コントロールデータの内容を取得(ok)
      call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      if(ihan.ne.0) then
      ierr_dat = 2
      call err_outf(ieer_dat)
      return
      endif
c――――― ★★解析制御情報をファイルから入力し、スレーブに転送するためバッファにセット
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★解析制御情報をファイルから入力(vpp)
      ALLOCATE (ff_data(400),if_data(400),iif_dt(2))
      iif_dt(1) = 0    ! ff_data の最終場所
      iif_dt(2) = 10   ! if_data の最終場所
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★動的解析ダイアログその１のデータを入力(ok)
      call dyctl1_pa(ierr,NINDIT,GINDIS,F1SEC,fs_st,fl_st,ifp_st,IST,
     +              JIKUZERO,G_JIKUZERO_ALPH,
     +              ff_data(iif_dt(1)+1),if_data(iif_dt(2)+1),iif_dt)
      if(ierr.ne.0) then
      ierr_dat = 3
      call err_outf(ieer_dat)
      endif
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★動的解析ダイアログその２のデータを入力(ok)
      call dyctl2_pa(ierr,NSTEP,F2SEC,DELT,IGRA,IBETA,BETA,GUMMA,XGAL,
     *            NNTIME,EPSDSP,load_memb_mass,dt_M_filter,IT_ANALYS,
     +            ff_data(iif_dt(1)+1),if_data(iif_dt(2)+1),iif_dt)
      if(ierr.ne.0) then
      ierr_dat = 4
      call err_outf(ieer_dat)
      endif
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★解析結果の出力パラメータを入力(ok)
      call doutcl_pa(ierr,IWSTP,SOUTSC,DMAXCK,No_section,
     +              ff_data(iif_dt(1)+1),if_data(iif_dt(2)+1),iif_dt)
      if(ierr.ne.0) then
      ierr_dat = 5
      call err_outf(ieer_dat)
      endif
c―――――――――――――――――――――――――――――――――――――――――――――――――――――★減衰ダイアログのデータを入力(ok)
      call damctl_pa(ierr,NREAD,ITYDP,NDMP,NDMP2,NHH,HH,QHH,
     +              ff_data(iif_dt(1)+1),if_data(iif_dt(2)+1),iif_dt)
      if(ierr.ne.0) then
```

```
            ierr_dat = 6
            call err_outf(ieer_dat)
          endif
c-------------------------------------------------★制御情報の取得に失敗した場合はここで戻る
         if(ierr_dat .ne. 0 ) then
          DEALLOCATE (ff_data,if_data,iif_dt)
          return
          endif
c-----------------------------------------------------------------------
c
c     ★         制御情報を構造体にセット
c
c-----------------------------------------------------------------------
          call Set_control(Control,N_analysis,NINDIT,GINDIS,
         *                 IWSTP,SOUTSC,DMAXCK,in_disp,in_stres,
         *                 IT_ANALYS,JIKUZERO,G_JIKUZERO_ALPH)        !ok (in_disp,in_stres  未定義)
c         write(damp_out,*) ' Set_control Ok'
          call Set_model_type(Model_type)
c         write(damp_out,*) ' Set_model_type Ok'
          call Set_newmark(Newmark_P,F1SEC,F2SEC,HH(1),HH(2),
         *                 QHH(1),QHH(2),DELT,BETA,EPSDSP,NNTIME,GUMMA,
         *                 ITYDP,NDMP,NDMP2)
c         write(damp_out,*) ' Set_newmark Ok'
          call Set_dynamic_load(Dynamic_load,IST,IGRA,
         *                      XGAL,load_memb_mass)
c         write(damp_out,*) ' Set_dynamic_load Ok',Control.jikuzero
c-----------------------------------------------------------------------
c
c     ★         構造データを予備入力し、構造用のパラメータを設定する(ok)
c
c-----------------------------------------------------------------------
          nfix=5
          nfi=1
          call infile(nfi,nfix,ierr)
          if(ierr.ne.0) then
          ierr_dat =10
          call err_outf(ieer_dat)
          return
          endif
          call Get_parameters(Parameter_C,ierr)
c         write(damp_out,*) ' Out Get_parameters'
          close(nfix)
          if(ierr .ne. 0) then
          ierr_dat =11
          call err_outf(ieer_dat)
          return
          endif
c-------------------------------------------------★制御情報をスレーブとモニターに転送(vpp)
c-------- ★★  制御情報と構造用制御情報を送信
          if(n_proc.ge.2) then
          call send_ctlset(Parameter_C,ff_data,if_data,iif_dt, N_analysis)
          endif
          DEALLOCATE (ff_data,if_data,iif_dt)
c-------------------------------------------------------------★地震加速度を予備入力(ok)
          call Get_earth_load(1,acc_earth,Dynamic_load,ierr)
          if(ierr.ne.0) then
          ierr_dat =21
          call err_outf(ieer_dat)
          return
          endif
c-------------------------------------------------------------★節点荷重履歴を予備入力(ok)
          call Get_point_load(1,fdd_point,Dynamic_load,ierr,
         *                    Newmark_P,fs_st,fl_st,ifp_st)
          if(ierr.ne.0) then
          ierr_dat =22
          call err_outf(ieer_dat)
          return
          endif
c-----------------------------------------------------------------------
c
c     ★    DLL を使用する場合、ワークエリアとして必要な大きさ及び DLL 番号を得る
c                                        DLL subroutine code No. 1
c
c-----------------------------------------------------------------------
c          if(Parameter_C.n_element_dll .ne. 0 )
c         * call Get_DLL_code(dll_code,n_request1,n_request2,
c         *                   n_request3,n_request4)
c-----------------------------------------------------------------------
c
c     ★         構造体の大きさを動的確保する（その１）
c
c-----------------------------------------------------------------------
          ALLOCATE (Max_disp(Parameter_C.n_point))
          ALLOCATE (Member(Parameter_C.n_member))
          ALLOCATE (Max_stress(Parameter_C.n_member))
          ALLOCATE (Element(Parameter_C.n_element))
          ALLOCATE (Point(Parameter_C.n_point))
c-----------------------------------------------------------------------
c
c     ★         新配列の大きさを動的確保する
c
c-----------------------------------------------------------------------
          N= Parameter_C.n_S_comp_model       ! 任意静的縮合型モデル
c         write(76,'(//a,i4)') ' Parameter_C.n_S_comp_model ',n
          if(N.ne.0) then
          nx=1
          call Send_S_comp_model(nx,nx_file,S_comp_model,Model_type ,ieerx,
         *                       id_buf,jd_buf,buf,ibuf)
```

```
      if(ieerx.ne.0) then
      write(76,*) ' 任意静的縮合型モデル用ファイルがありません ',ieerx
      Parameter_C.n_S_comp_model=0
      return
      else
      N = Parameter_C.nE_New_Element     !  任意型静的縮合モデルに含まれる要素エレメント数
c     write(76,'(a,i4)') ' Parameter_C.nE_New_Element ',n
      if(N.ne.0)then
      ALLOCATE (E_Fiber_work(N))         ! 新規縮合モデルの要素エレメント数の動的確保  ①-2
      endif
      N = nx_file                        ! 任意静的縮合型モデル
      Parameter_C.n_S_comp_model=N
c     write(76,'(a,i4)') ' nx_file of S_comp_model',n
      if(N.ne.0)then
      ALLOCATE (S_comp_model(N))         ! 新規縮合モデル設定領域の動的確保
      endif
c     write(76,'(/a,i4)') ' Get_S_comp_model in '
      nx=2
      call Send_S_comp_model(nx,nx_file,S_comp_model,Model_type ,ieerx,
     *                       id_buf,jd_buf,buf,ibuf)
c     write(76,*) ' Get_S_comp_model in ',id_buf,jd_buf
      ALLOCATE (buf(id_buf+10),ibuf(jd_buf+10))
      nx=3
      call Send_S_comp_model(nx,nx_file,S_comp_model,Model_type ,ieerx,
     *                       id_buf,jd_buf,buf,ibuf)
      DEALLOCATE (buf,ibuf)
      endif
      endif
c-----------------------------------------------------------------------
c
c     ★         配列の大きさを動的確保する
c
c-----------------------------------------------------------------------
      N=Parameter_C.n_point              ! 節点数
      ALLOCATE (
     *   fll_static_point(3,6,N),am_point(2,N),
     *   fll_force_point(3,N)
     *         )
      N=Parameter_C.n_member             ! 部材数
      ALLOCATE (
     *   am_member(12,12,N),
     *   rot_memb_t(3,3,N),rot_memb(3,3,2,N),
     *   ak_linear(12,12,N),ak_nonlinear(12,12,N)
     *         )
      ALLOCATE (pa_work_force(33,N))  ! 部材応力転送用バッファ

      N=Parameter_C.n_local_coord        ! 局所座標系を使用する場合
      if(N.ne.0)ALLOCATE (
     *   rot_local(3,3,N)
     *)
      M_alloc(1)=1
c-----------------------------------------------------------------------
c
c     ★         構造・荷重データを入力し、データの設定を行う
c
c-----------------------------------------------------------------------
c-----------------------------------------------★基本構造データを入力(ok)
      nfix=5
      nfi=1
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat =10
      call err_outf(ieer_dat)
      return
      endif
c-------- ★★  構造データの送信用バッファ領域を確保
c     バッファデータ仕様
c     buf()     1：座標 3
c               2：局所座標 3
c               3：要素 17
c               4：部材 4
c     ibuf()    1：境界拘束条件 7
c               2：要素 6
c               3：部材  14
c
      id_buf=Parameter_C.n_point*6+Parameter_C.n_element*17+
     *       Parameter_C.n_member*4
      jd_buf=Parameter_C.n_point*7++Parameter_C.n_element*6+
     *       Parameter_C.n_member*14
c     write(damp_out, '(3i5)') Parameter_C.n_point,
c     +      Parameter_C.n_element, Parameter_C.n_member
      ALLOCATE (buf(id_buf+10),ibuf(jd_buf+10))
c-------- ★★  構造データの送信用バッファ領域を確保
      call Get_structure_pa(Point,Member,Element,Parameter_C,
     *                  Model_type,ierr,buf,ibuf,id_buf,jd_buf,
     *                  S_comp_model,E_Fiber_work)
      close(nfix)
      if(ierr .ne. 0) then
      ierr_dat =iabs(ierr)
c                         err No. 12-19 使用
      call err_outf(ieer_dat)
      DEALLOCATE (buf,ibuf)
      return
      endif
c-----------------------------------------------------------------------
c
c     ★         E_Fiber_work の内容を別途送信
c
```

```
c-----------------------------------------------------------------------
      N= Parameter_C.n_S_comp_model       ! 任意静的縮合型モデル
c     write(76,*) ' Parameter_C.n_S_comp_model ',n
      if(n_proc.ge.2) then
      if(N.ne.0) then
      call send_E_fiber(E_Fiber_work,S_comp_model,
     *                                Element,Parameter_C)
      endif
c------- ★★  スレーブ、モニターに構造データを送信
      call Send_structure(buf,ibuf,id_buf,jd_buf)
      endif
      DEALLOCATE (buf,ibuf)
c     write(damp_out,*) ' Get_structure ok'
c-----------------------------------------------------------------------
c
c     ★         制震セミアクティブダンパー用フィルターのパラメータセット
c
c-----------------------------------------------------------------------
      if(Model_type.n_m_model(6) .ne. 0) then
      call Set_Maxwell_filter(dt_M_filter,Newmark_P.dt,Model_type,ierr)
      if(ierr.ne.0) then
      ierr_dat = 20
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Set_Maxwell_filter Ok'
      endif
c-----------------------------------------------------------------------
c
c     ★         計算用構造体の大きさを動的確保する（その２）
c
c-----------------------------------------------------------------------
      N=Model_type.n_m_damp
      if(N.ne.0)then
      ALLOCATE (ac_member(12,12,N))
      endif
c-------------------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c-------------------------------------------------★Model_No.2 3 次元せん断弾塑性モデル
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      ALLOCATE (RO_work(n))
      endif
c-------------------------------------------------★Model_No.3 3 次元軸力弾塑性モデル
c-------------------------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
c-------------------------------------------------★Model_No.5 3 次元免振モデル
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      ALLOCATE (MSS_work(n))
      endif
c-------------------------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      n=Model_type.n_m_model(6)
      if(n.ne.0) then
      ALLOCATE (E_model6_real(n))
      endif
c-------------------------------------------------★Model_No.7 3 次元バネモデル
c      n=Model_type.n_m_model(7)
c     if(n.ne.0) then
c       ALLOCATE (E_model7(n))
c      endif
c-------------------------------------------------★ファイバーモデル
c-------------------------------------------------★Model_No.11 両端ファイバーモデル
      n= Model_type.n_e_model(11)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model11(n))
      endif
      n= Model_type.n_m_model(11)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model11(n))
      endif
c-------------------------------------------------★Model_No.12 両端、中央ファイバーモデル
      n= Model_type.n_e_model(12)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model12(n))
      endif
      n= Model_type.n_m_model(12)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model12(n))
      endif
c-------------------------------------------------★Model_No.13 両端、中央ファイバーモデル
      n= Model_type.n_e_model(18)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model13(n))
      endif
      n= Model_type.n_m_model(18)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model13(n))
      endif
c-------------------------------------------------★Model_No.15 両端ファイバーFEM モデル
      n= Model_type.n_e_model(15)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model15(n))
      endif
      n= Model_type.n_m_model(15)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model15(n))
      endif
c-------------------------------------------------★Model_No.21 両端 MS モデル
      n= Model_type.n_e_model(13)         !要素モデルの数
```

```
      if(n.ne.0) then
      ALLOCATE (E_model21(n))
      endif
      n= Model_type.n_m_model(13)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model21(n))
      endif
c-----------------------------------------------★Model_No.22 両端、中央 MS モデル
      n= Model_type.n_e_model(14)        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model22(n))
      endif
      n= Model_type.n_m_model(14)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model22(n))
      endif
c-----------------------------------------------★Model_No.31 両端アナロジーモデル
      n= Model_type.n_e_model(16)        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model31(n))
      endif
      n= Model_type.n_m_model(16)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model31(n))
      endif
c     write(damp_out,*) ' allocate model 31 ',n
c-----------------------------------------------★Model_No.32 両端、中央アナロジーモデル
      n= Model_type.n_e_model(17)        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model32(n))
      endif
      n= Model_type.n_m_model(17)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model32(n))
      endif
c     write(damp_out,*) ' allocate model 32 ',n
c-----------------------------------------------★Model_No.32 両端、中央アナロジーモデル
      n= Model_type.n_e_model(19)        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model33(n))
      endif
      n= Model_type.n_m_model(19)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model33(n))
      endif
c     write(damp_out,*) ' allocate model 32 ',n
c-----------------------------------------------★新規モデル Model_No.51-70
      n= Model_type.n_e_New_fiber        !要素モデルの数
c     write(76,'(a,i4)') 'Model_type.n_e_New_fiber',n
      if(n.ne.0) then
      ALLOCATE (E_modelx(n))                                          ! ②-2
      Endif
      n= Model_type.n_m_New_fiber        ! 部材モデルの数
c     write(76,'(a,i4)') 'Model_type.n_m_New_fiber',n
      if(n.ne.0) then
      ALLOCATE (M_modelx(n))                                          ! ④-2
      endif
      n=Parameter_C.nM_New_Element       ! 数新規モデルにおける部材エレメント数
c     write(76,'(a,i4)') 'Parameter_C.nM_New_Element',n
      if(n.ne.0) then
      ALLOCATE (M_Fiber_work(n))                                      ! ⑤-2
      endif
c-----------------------------------------------★Model_No.51 3 次元プレテンション動作モデル
c      n=Model_type.n_m_model(51)
c     if(n.ne.0) then
c      ALLOCATE (M_model51(n),
c     *          E_model51(n))
c      endif
c-----------------------------------------------------------------------
c
c     ★         配列の大きさを動的確保する（その２：DLL 用）
c
c-----------------------------------------------------------------------
c     if(Parameter_C.n_element_dll .ne. 0 ) then
c      N1=n_request1* Parameter_C.n_element_dll
c      N2=n_request2* Parameter_C.n_element_dll
c      N3=n_request3* Parameter_C.n_member_dll
c      N4=n_request4* Parameter_C.n_member_dll
c      ALLOCATE (
c     *   work1_elememt(N1),work2_elememt(N2),
c     *   work1_member(N3),work2_member(N4)
c     *         )
c     endif
c-----------------------------------------------★節点荷重、加速度データ領域を確保
      if(Dynamic_load.n_load_dynamic .ne. 0) then
      ALLOCATE (acc_earth(3,Dynamic_load.n_load_dynamic))
      endif
      if(Dynamic_load.n_load_point .ne. 0) then
      ALLOCATE (fdd_point(3,Dynamic_load.n_load_point))
      endif
      M_alloc(2)=1
c-----------------------------------------------------------------------
c
c     ★         構造・荷重データを入力し、データの設定を行う（その２）
c
c-----------------------------------------------------------------------
c-----------------------------------------------★地震荷重を入力(ok)
      if(Dynamic_load.load_dynamic(1)  .ne. 0 .or.
```

```
     *   Dynamic_load.load_dynamic(2)  .ne. 0 .or.
     *   Dynamic_load.load_dynamic(3)  .ne. 0 ) then
      call Get_earth_load(2,acc_earth,Dynamic_load,ierr)
      if(ierr.ne.0) then
      ierr_dat =21
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Get_earth_load Ok'
      endif
c-------------------------------------------------★節点荷重分布を入力(ok)
      if(Dynamic_load.load_point(1)  .ne. 0 .or.
     *   Dynamic_load.load_point(2)  .ne. 0 .or.
     *   Dynamic_load.load_point(3)  .ne. 0 ) then
      call Get_point_loadf(fll_static_point,Parameter_C,
     *                     Dynamic_load,ierr)
      if(ierr.ne.0) then
      ierr_dat =22
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Get_point_loadf Ok'
c-------------------------------------★節点荷重時刻歴を入力およびセット
      call Get_point_load(2,fdd_point,Dynamic_load,ierr,
     *                    Newmark_P,fs_st,fl_st,ifp_st)
      if(ierr.ne.0) then
      ierr_dat = 23
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Get_point_load Ok'
      endif
c-------------------------------------------------★初期不整データを入力(ok)
      if(Control.init_imperfection .ne. 0) then
      nfix=5
      nfi=16
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 24
      call err_outf(ieer_dat)
      return
      endif
      ihan=0
      call Get_imperfection_pa(ihan,Control.amp_imperfection,Point,
     *     Parameter_C,buf,npoint_imp,ii1_buf)
      ihan=1
      id_buf=npoint_imp*4+1
c     write(damp_out, '(a, i5, a, i5)')
c     +     "npoint_imp=", npoint_imp, " id_buf=", id_buf
      ALLOCATE (buf(id_buf+10))
      call Get_imperfection_pa(ihan,Control.amp_imperfection,Point,
     *     Parameter_C,buf,npoint_imp,ii1_buf)
c------- ★★  スレーブに初期不整データを転送
      if(n_proc.ge.2) then
      call Send_imperfection(buf,ii1_buf,npoint_imp)
      endif
      DEALLOCATE (buf)
      close(nfix)
      endif
c-------------------------------------------------★システム内要素データを入力
c     ★         ファイバーデータ
c-------------------------------------★ファイバーデータの予備入力(ok)
      n_element=Model_type.n_e_model(11)+Model_type.n_e_model(12)+
     *          Model_type.n_e_model(15)+
     *          Model_type.n_e_model(14)+Model_type.n_e_model(13)+
     *          Model_type.n_e_model(16)+Model_type.n_e_model(17)+
     *          Model_type.n_e_model(18)+Model_type.n_e_model(19)+
     *          Model_type.n_e_New_fiber

      if(n_element .ne. 0) then
      nfix=5
      nfi=54
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 25
      call err_outf(ieer_dat)
      return
      endif
      call Fiber_input_pa(0,ierr,Parameter_C.n_member,
     *     Parameter_C.n_element,Member,Element,Model_type,
     *     E_model_fiber,M_model_fiber,E_model11,M_model11,
     *     E_model12,M_model12,E_model13,M_model13,
     *     E_model15,M_model15,
     *     E_model21,M_model21,E_model22,M_model22,
     *     E_model31,M_model31,E_model32,M_model32,
     *     E_model33,M_model33,
     *     id_buf,jd_buf,buf,ibuf,
     *     M_Fiber_work, E_Fiber_work,E_modelx,M_modelx,Parameter_C)
      close(nfix)
      if(ierr.ne.0)then
      ierr_dat = 26
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Fiber_input ok',id_buf,jd_buf
c------- ★★スレーブにファイバー用バッファをゼロセット
      ALLOCATE (buf(id_buf+10),ibuf(jd_buf+10))
      do i=1,id_buf+10
```

```
      buf(i)=0.
      enddo
      do i=1,jd_buf+10
      ibuf(i)=0
      enddo
c-------------------------------------★ファイバーモデル領域セット
      n= Model_type.nm_div_fmodel          !要素モデル内のサブ要素の数
      if(n.ne.0) then
      ALLOCATE (M_model_fiber(n))
      endif
      n= Model_type.nm_div_felement        !ファイバー要素のエレメント最大数
      if(n.ne.0) then
      ALLOCATE (E_model_fiber(n))
      endif
      M_alloc(3)=1
c-------------------------------------★ファイバーモデルデータ入力
      nfix=5
      nfi=54
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 25
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Fiber_input in 2'
      call Fiber_input_pa(1,ierr,Parameter_C.n_member,
     *      Parameter_C.n_element,Member,Element,Model_type,
     *      E_model_fiber,M_model_fiber,E_model11,M_model11,
     *      E_model12,M_model12,E_model13,M_model13,
     *      E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,
     *      E_model33,M_model33,
     *      id_buf,jd_buf ,buf,ibuf,
     *      M_Fiber_work, E_Fiber_work,E_modelx,M_modelx,Parameter_C)

      close(nfix)
      if(ierr.ne.0) then
      ierr_dat = 26
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Fiber_input ok'
c------- ★★スレーブにファイバーデータを転送
      if(n_proc.ge.2) then
      call Send_Fiber(id_buf,jd_buf ,buf,ibuf)
      endif
      DEALLOCATE (buf,ibuf)
      endif
c     write(damp_out,*) ' Fiber_send ok'
c-------------------------------------★修正 RO モデル領域セット
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      nfix=5
      nfi=53
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 35
      call err_outf(ieer_dat)
      return
      endif
      call RO_data_input_pa(0,n,RO_work,Element,Parameter_C.n_element,
     *      ierr,id_buf,jd_buf,buf,ibuf )
      if(ierr.ne.0) then
      ierr_dat = 36
      call err_outf(ieer_dat)
      return
      endif
      ALLOCATE (buf(id_buf+10))
      call RO_data_input_pa(1,n,RO_work,Element,Parameter_C.n_element,
     *      ierr,id_buf,jd_buf,buf,ibuf )
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat = 36
      call err_outf(ieer_dat)
      DEALLOCATE (buf)
      return
      endif
c     write(damp_out,*) ' RO_data_input ok'
c------- ★★スレーブに RO モデル用データを転送
      if(n_proc.ge.2) then
      call Send_RO_data(id_buf,jd_buf,buf)
      endif
      DEALLOCATE (buf)
      endif
c-------------------------------------------------★DLL 内要素データを入力
c                                        DLL subroutine code No. 2
c      n_element=Parameter_C.n_member_dll
c      if(n_element .ne. 0) then
c      nfix=5
c     nfi=60
c      call infile(nfi,nfix,ierr)
c     if(ierr.ne.0) then
c      ierr_dat =13
c     return
c     endif
c      call Get_element_dll(work1_elememt,work2_elememt,Parameter_C)
c     close(nfix)
```

```
c     endif
c-------------------------------------------------★質量データを入力(ok)
      nfix=5
      nfi=2
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 27
      call err_outf(ieer_dat)
      return
      endif
      call Get_mass(ierr,Point,Parameter_C)
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat = 28
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Get_mass Ok'
c-------------------------------------------------★レーリー減衰を入力(ok)
      nfix=5
      nfi=43
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 29
      call err_outf(ieer_dat)
      return
      endif
      call Get_damp(Newmark_P,ierr)
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat =30
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Get_damp Ok'
c------- ★★スレーブに減衰用データを転送
      if(n_proc.ge.2) then
      call Send_damp(Newmark_P,ierr)
      endif
c-----------------------------------------------------------------------
c
c     ★         動的解析のための予備計算を行う
c
c-----------------------------------------------------------------------
c------- ★★  Time Check Codes 予備計算
      call MPI_WTime(TIME_PRE, nStart)
c-------------------------------------------------★構造物の不安定性チェック
c      call check_structure(Point,Element,Parameter_C)
c-------------------------------------------------★部材長さ計算(ok)
      call Cal_member_length(Member,Point,Parameter_C)
c     write(damp_out,*) ' Cal_member_length Ok'
c-------------------------------------------------★モデルの初期設定
      call Set_initial_data(Element,Member,Parameter_C,Newmark_P,
     *            E_model6_real,Model_type)
c     write(damp_out,*) ' Set_initial_data Ok'
c-------------------------------------------------★DLL 非線形エリアの設定
c      if(Parameter_C.n_member_dll .ne. 0) then
c      call Set_nonlinear_S(Element,Parameter_C,
c     *     work1_elememt,work2_elememt,work1_member,work2_member)
c      endif
c-------------------------------------------------★座標変換行列計算
      call Get_rotate_all(rot_memb_t,Parameter_C,Point,Member)
c               write(damp_out,*) ' Get_rotate_all Ok'
c-------------------------------------------------★局所座標変換行列計算
      call Get_rot_local(rot_local,Parameter_C,Point)
c               write(damp_out,*) ' Get_rot_local Ok'
c-------------------------------------------------★釣合座標系標変換行列計算
      call Get_rotate(rot_memb,rot_memb_t,rot_local,
     *                Parameter_C,Member)
c               write(damp_out,*) ' Get_rotate Ok'
c-------------------------------------------------★節点拘束表の作成
c                                                   未知数等をセット
      call Set_restraint_point(Parameter_C,Point,Control)
c               write(damp_out,*) ' Set_restraint_point Ok'
c-----------------------------------------------------------------------
c
c     ★         配列の大きさを動的確保する（その３）
c
c-----------------------------------------------------------------------
      N=Parameter_C.n_unknown
c      ALLOCATE (test_vector(N))
      if(n_proc.ge.2)then
      ALLOCATE (Ms_table(N,n_proc-1),Ms_free(n_proc-1))
      ALLOCATE (buf_disp(N+1))
      endif
      ALLOCATE (
     *     disp_point(N),vel_point(N),acc_point(N),
     *     est_disp_point(N),est_vel_point(N),
     *     est_ddisp_point(N)
     *         )
      ALLOCATE (
     *   result_disp_point(N),result_vel_point(N),result_acc_point(N),
     *   past_disp_point(N),past_vel_point(N),past_acc_point(N),
     *   past_dacc_point(N)
     *         )
      ALLOCATE (
     *   ld_point(N),ld_point_repeat(N),fld_static(3,N),
     *   a_vector(N),b_vector(N)
```

```
     *         )
      ALLOCATE (max_h_sky(0:N+1)
     *         )
      M_alloc(4)=1
c-------------------------------------------------★部材両端の拘束表作成(ok)
      call Set_restraint_member(Parameter_C,Member,Point)
c      write(damp_out,*) ' Set_restraint_member Ok'
c-------------------------------------------------★スカイライン変換表作成(ok)
c-------------------------------------------------★スカイライン行列の領域数等をセット
      call Cal_table_sky(max_h_sky,Parameter_C,Member)
c     write(damp_out,*) ' Cal_table_sky Ok'
c-----------------------------------------------------------------------
c
c     ★         配列の大きさ（スカイライン用）を動的確保する（その４）
c
c-----------------------------------------------------------------------
c-------------------------------------------------★ファイバー用履歴要素
      n= Model_type.n_m_bilinear          !  バイリニアの履歴要の数
      if(n.ne.0) then
      ALLOCATE (Bilinear_work(n))
      endif
      n= Model_type.n_m_trilinear          !  トリリニアの履歴要の数
      if(n.ne.0) then
      ALLOCATE (Trilinear_work(n))
      endif
      n= Model_type.n_m_Concrete          !  コンクリートの履歴要の数
      if(n.ne.0) then
      ALLOCATE (Concrete_work(n))
      endif
c-------------------------------------------------★全体剛性行列など
      ALLOCATE (gskym(Parameter_C.n_skyline))
      ALLOCATE (gskym_d(Parameter_C.n_unknown))
      ALLOCATE (nwork(Parameter_C.n_unknown))
      ALLOCATE (twork(Parameter_C.n_unknown))
      M_alloc(5)=1
c---------------------------------------★部材両端中央応力、力のゼロセット(ok)
      call  Set_pointforce_zero(Member,Parameter_C.n_member)
c     write(damp_out,*) ' Set_pointforce_zero Ok'
c---------------------------------------★部材応力のゼロセット(ok)
      call  Set_stress_zero(Member,Parameter_C.n_member)
c     write(damp_out,*) ' Set_stress_zero Ok'
c-------------------------------------------------★MSS モデルの初期設定
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      call Cal_MSS_dat(Member,Element,Model_type,
     *                 MSS_work,Parameter_C)
c     write(damp_out,*) ' Cal_MSS_dat Ok'
      endif
c-------------------------------------------------★平面問題における部材の拘束方向チェック(ok)
      call Check_R_direction(Control.analysis_3D,Parameter_C,
     *                       Member,rot_memb)
c     write(damp_out,*) ' Check_R_direction ok'
c-------------------------------------------------★部材の線形剛性計算(ok)
      call Cal_stiff_linear(Model_type,Element,Member,Parameter_C,
     *      ak_linear,E_model11, E_model_fiber,M_model11, M_model_fiber,
     *      E_model12,M_model12,E_model13,M_model13,E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,E_model33,M_model33,
     *      Bilinear_work,Trilinear_work,Concrete_work,
     *      work1_elememt,work2_elememt,work1_member,work2_member,
     *      S_comp_model, E_modelx, M_modelx,
     *      E_fiber_work, M_fiber_work)
c     write(damp_out,*) ' Cal_stiff_linear Ok'
c-------------------------------------------------★剛性の釣合座標系への変換(ok)
      call Rotate_stiffness(Parameter_C,ak_linear,rot_memb)
c     write(damp_out,*) ' Rotate_stiffness Ok'
c-------------------------------------------------★部材の減衰行列計算(ok)
      if(Parameter_C.nc_member .ne. 0) then
      call Cal_damp_linear(Element,Member,Parameter_C,ac_member,
     *            E_model6_real,work1_elememt,
     *            work2_elememt,work1_member,work2_member)
c     write(damp_out,*) ' Cal_damp_linear Ok'
c-------------------------------------------------★部材減衰行列の釣合座標系への変換(ok)
      call Rotate_damp(Parameter_C.n_member,ac_member,rot_memb,Member)
c     write(damp_out,*) ' Rotate_damp Ok'
      end if
c-------------------------------------------------★節点集中質量セット(ok)
      call Set_mass(Point,Parameter_C,am_point)
c     write(damp_out,*) ' Set_mass Ok'
c-------------------------------------------------★節点荷重セット(ok)
      call Set_point_load(fll_static_point,Parameter_C,
     *              Dynamic_load,Point,fld_static,rot_local)
c     write(damp_out,*) ' Set_point_load Ok'
c-------------------------------------------------★接線剛性のコピー(ok)
      call Initset_nonlin_stiff(ak_linear,ak_nonlinear,
     *                          Parameter_C.n_member)
c     write(damp_out,*) ' Initset_nonlin_stiff Ok'
c-------------------------------------------------★ベクトルのゼロセット
c-----------------------------------------★増分前の変位、速度、加速度(ok)
      call Set_zero_v(past_disp_point, past_vel_point,
     *       past_acc_point,past_dacc_point,Parameter_C.n_unknown)
c     write(damp_out,*) ' Set_zero_v Ok'
c---------------------------------------★最大変位等のゼロセット(ok)
      call Clear_max_disp(Max_disp,Parameter_C.n_point)
c     write(damp_out,*) ' Clear_max_disp Ok'
c---------------------------------------★最大、最小応力等のゼロセット(ok)
      call Clear_max_stress(Max_stress,Parameter_C.n_member)
c     write(damp_out,*) ' Clear_max_stress Ok'
```

```
c-----------------------------------------------------------------------
c
c     ★          静的解析結果等を取り込む
c                （初期変位、初期応力を入力）
c
c-----------------------------------------------------------------------
c-------------------------------------------------★初期変位を入力
      if(Control.init_disp.ne.0) then
      nfix=5
      nfi=16
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 31
      call err_outf(ieer_dat)
      return
      endif
c       call Get_init_disp(Point,Member,Parameter_C,ierr)
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat =32
      call err_outf(ieer_dat)
      return
      endif
c     write(damp_out,*) ' Get_damp Ok'
c------- ★★スレーブに初期変位を転送
c     call Send_init_disp(Point,Member,Parameter_C,ierr,MPP_Ana_Group)
      endif
c-------------------------------------------------★初期応力を入力
      if(Control.init_stress.ne.0) then
      nfix=5
      nfi=52
c      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat = 33
      call err_outf(ieer_dat)
      return
      endif
c      call Get_init_stress(Point,Member,Parameter_C,ierr)
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat =34
      call err_outf(ieer_dat)
      return
      endif
      write(damp_out,*) ' Get_init_stress Ok'
c------- ★★スレーブに初期応力を転送
c     call Send_init_stress(Point,Member,Parameter_C,ierr,MPP_Ana_Group)
      endif
c-----------------------------------------------------------------------
c
c     ★          解析結果を出力するファイル群をオープンする(ok)
c
c-----------------------------------------------------------------------
      call flcheck(ifl,ifly,iflz,ierr,Control.type_analysis)
      if(ierr.ne.0) then
      ierr_dat =14
      endif
c-------------------------------------------------★出力指定した断面がファイバー要素
c-------------------------------------------------★かどうかチェック(ok)
       call out_section_check_pa(Member,Element,
     *                Parameter_C.n_member,No_section,Out_section,
     *                ifl,iflz,i_print,n_member1,n_member2,S_comp_model
     *                ,E_modelx,M_modelx,E_fiber_work,M_fiber_work)
c-------------------------------------------------★ファイル名一覧出力
      write(damp_out,'(//a,i5/)') ' 出力ファイル名一覧 '
     *  ,Dynamic_load.load_dynamic(1)
      do i=1,16
      write(damp_out,'(i4,3i6)') i,ifl(i),iflz(i),ifly(i)
      enddo
c-----------------------------------------------------------------------
c
c     ★          描画用データのセット
c
c-----------------------------------------------------------------------
c-----------------------------------------★変位
      if(i_read_disp .ne. 0) then
c------- ★★モニターの描画用データを作成
      call Set_preset_disp(0,n_point,past_disp_point,F_disp,Point,
     *                           rot_local,Parameter_C)
c       write(damp_out,*) ' Set_preset_disp ok',n_proc
      endif
c-----------------------------------------------------------------------
c
c     ★          解析ステップをセットする
c
c-----------------------------------------------------------------------
      ns_step=1
      n_step=10
      d_max_v=0.
      id_max_v=0
c-------------------------------------------------★解析手法のセット
      N_implicit_method = 1  !陰解法は１回で反復法に戻る
      Iteration_method  = 1  !最初は反復法を使用
c------- ★★加速度のスレーブ転送用テーブル作成
c       write(damp_out,*) ' Convert_node_inf_vpp out Ok',n_proc
      if(n_proc.ge.2) then
c     write(damp_out,'(//a,i4)') ' Convert_node_inf_vpp in Ok',n_unknown
      call Convert_node_inf_vpp(Parameter_C.n_unknown,Parameter_C,
```

```
     *       Member,Point,Ms_table,Ms_free,n_proc,nm_parallel)
c     write(damp_out,*) ' Convert_node_inf_vpp out Ok'

c     do i=1, n_proc-1
c          write(damp_out,'(a,i3,a,i10)') 'ms_free(',i,')=',Ms_free(i)
c     end do
      endif
c-----------------------------------------------------------------------
c-------------------------------------★予備計算時間のセット
c------- ★★  Time Check Codes 予備計算
      call MPI_WTime(TIME_PRE, dTime_Pre)
      All_Time_Pre = All_Time_Pre + dTime_Pre
      stimes=0
      S_control_code = 0                  ! スレーブコントロールコードの初期設定
c-----------------------------------------------------------------------
c
c     ★         予備計算はここで終了 (GUIに戻る)
c
c-----------------------------------------------------------------------
c      write(damp_out,*) ' return ',Control.jikuzero
      return
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
c     ★         動的解析開始
c
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
      i_print = 0
 9990 continue
c------- ★★  総解析時間
      if(ns_step.le.1) call MPI_WTime(TIME_ALL, nStart)              ! 解析時間を計る
      all_net3 = 0
      all_net4 = 0
      n_iterate = 0                          !  反復回数ゼロセット
      n_member=Parameter_C.n_member
      n_point =Parameter_C.n_point
      n_unknown=Parameter_C.n_unknown
      n_local_coord=Parameter_C.n_local_coord
      write(damp_out,'(//a,4i4)') ' Dynamic analysis start :',
     *                 ns_step,n_step,n_unknown
      do 9999 istep=ns_step,ns_step+n_step - 1
c-------------------------------------★解析開始コードをスレーブに送る
c------- ★★スレーブに解析開始コードを転送
c          スレーブ制御コード 0:  収束
c                             1:  未収束
c                             2:  構造物崩壊もしくはエラーは発生、途中終了
c          if ( n_proc .ge. 2)then
c------- ★★  Time Check Codes
c     call MPI_WTime(TIME_NET_CONV, nStart)
c          ite_end = 0
c          call bcast_ite(ite_end)
c------- ★★  Time Check Codes
c                call MPI_WTime(TIME_NET_CONV, dTime_NET_CONV)
c                All_Time_NET_CONV = All_Time_NET_CONV + dTime_NET_CONV    ! 収束コード通信
c          endif
c------- ★★  Time Check Codes
c     write(damp_out,*) "動的解析"
      call MPI_WTime(TIME_STEP, nStart)
      T=Newmark_P.dt*istep
      i_print=mod(istep,Control.interval_out)
C-------------------------------------★解析ステップの出力
      write(damp_out,'(/a,i6,a,f10.3,a,f12.4,a,i4,a,f12.8)')
     *     ' Step No,:',istep,
     *     '     Time :',T,
     *     'sec.     Max. Disp. :',d_max_v,
     *     '     Node Number :',id_max_v,
     *     ' times (s):',dTime_STEP/1000000.
c-----------------------------------------------------------------------
c
c     ★         第一と第二ステップの最初にスカイライン行列を作成、分解する
c
c-----------------------------------------------------------------------
c-------------------------------------★左辺係数行列の計算(ok)
c-------------------------------------★ステップ番号のセット(ok)
c      write(76,'(a,i4)') ' 解析開始',istep
      if(istep.eq.1.or.istep.eq.Newmark_P.n2_step) then
c------- ★★  Time Check Codes 係数行列作成
      call MPI_WTime(TIME_K, nStart)
      n_istep=1
      if(istep.eq.Newmark_P.n2_step) n_istep=2
c-------------------------------------★スカイライン行列のゼロセット(ok)
      n_skyline=Parameter_C.n_skyline
c      write(damp_out,*) ' Set_sky_zero in',n_skyline
      call Set_sky_zero(gskym,n_skyline)
c      write(damp_out,*) ' Set_sky_zero ok'
c-------------------------------------★集中質量系の行列への組み込み
c-------------------------------------★レーリー減衰を含む
      call Build_sky_mm(n_istep,gskym,
     *           Point,n_point, am_point , rot_local,
     *           n_local_coord ,Newmark_P, max_h_sky)
c       write(damp_out,*) ' Build_sky_mm ok'
c-------------------------------------★部材の整合質量系の行列への組み込み(ok)
c-------------------------------------★レーリー減衰を含む
c-------------------------------------★部材の整合質量行列計算(ok*)
      if(Dynamic_load.load_mass .ne. 0) then
      call Cal_mass_linear(n_istep,Element,Member,Parameter_C,am_member,
     *           work1_elememt,work2_elememt,work1_member,work2_member,
```

```
     *            Dynamic_load.load_mass)
c       write(damp_out,*) ' Cal_mass_linear ok'
c-------------------------------------★整合質量の釣合座標系への変換(ok*)
      call Rotate_mass(n_istep,Element,Member,n_member,am_member,
     *                 rot_memb,Dynamic_load.load_mass)
c       write(damp_out,*) ' Rotate_mass ok'
c-------------------------------------★整合質量系の組み込み(ok*)
      call Build_sky_m(n_istep,gskym,n_skyline, Member,n_member,
     *                  am_member ,Newmark_P, max_h_sky,Element)
      endif
c       write(damp_out,*) ' Build_sky_m ok'
c-------------------------------------★部材減衰系の組み込み(ok)
      if(Parameter_C.nc_member .ne. 0) then
      call Build_sky_c(gskym,Member,n_member,
     *                 ac_member ,Newmark_P, max_h_sky)
c        write(damp_out,*) ' Build_sky_c ok'
      endif
c---------------------------------------★線形剛性の組み込み(ok)
c---------------------------------------★レーリー減衰を含む
      call Build_sky_k(n_istep,gskym,n_skyline, Member,n_member,
     *                       Ak_linear, Newmark_P, max_h_sky)
c        write(damp_out,*) ' Build_sky_k ok'
c-----------------------------------------★行列の LDU 分解(ok)
      n_skyline=Parameter_C.n_skyline
      call decomp_sky(n_skyline,n_unknown,n_unknown,max_h_sky,
     *                gskym,gskym_d,nwork,twork,iexit)
c      write(damp_out,'(a,12f12.3)')' nwork'
c-----------------------------------------★分解成功か？
      if(iexit.ne.0) then
      ierr_dat =100
      write(damp_out,*) ' decomp_sky err',iexit
      return
      endif
c      write(damp_out,*) ' decomp_sky ok',iexit
c---------------------------------------★解析時間のセット
c-------- ★★  Time Check Codes 係数行列作成
      call MPI_WTime(TIME_K, dTime_K)
      All_Time_K = All_Time_K + dTime_K !  係数行列作成
      endif
c-----------------------------------------------------------------------
c
c    ★        動的解析の開始
c
c-----------------------------------------------------------------------
      if(Control.type_analysis .ne. 7) goto 9981
c-----------------------------------------------------------------------
c
c    ★        線形解析
c
c-----------------------------------------------------------------------
c-------- ★★  Time Check Codes 右辺項作成
      call MPI_WTime(TIME_LD, nStart)
c---------------------------------------★右辺の定数ベクトルゼロセット(ok)
      call Clear_vec(n_unknown,ld_point )
c         write(damp_out,*) ' Clear_vec ok'
c---------------------------------------★地震加速度セット(ok)
      acc1=Get_Acc(T,1,acc_earth,Dynamic_load,Newmark_P.f1_T)
      acc2=Get_Acc(T,2,acc_earth,Dynamic_load,Newmark_P.f1_T)
      acc3=Get_Acc(T,3,acc_earth,Dynamic_load,Newmark_P.f1_T)
c        write(damp_out,'(a,4f10.3)') ' Get_Acc ok'
c---------------------------------------★集中質量に関する慣性項
      call Add_earth1_ld(n_istep,acc1,acc2,acc3,ld_point,
     *           Point,n_point,am_point,rot_local,Parameter_C)
c      write(damp_out,*) ' Add_earth1_ld ok'
c---------------------------------------★整合質量に関する慣性項
      call Add_earth2_ld(acc1,acc2,acc3,ld_point,
     *    Member,n_member,am_member,rot_local,Parameter_C,Dynamic_load)
c         write(damp_out,*) ' Add_earth2_ld ok'
c---------------------------------------★節点荷重のセット(ok)
      p1=Get_Ps(T,1,fdd_point,Dynamic_load)
      p2=Get_Ps(T,2,fdd_point,Dynamic_load)
      p3=Get_Ps(T,3,fdd_point,Dynamic_load)
c       write(damp_out,'(a,4f10.3)') ' Get_Ps ok'
      call Add_point_ld(p1,p2,p3,ld_point,n_unknown,
     *       Dynamic_load,fld_static)
c     write(damp_out,*) ' Add_point_ld ok'
c---------------------------------------★線形減衰項計算(ok)
c---------------------------------------★集中質量(ok)
      call Add_damp1_ld(n_istep,ld_point,Point,n_point,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       am_point,Newmark_P,Parameter_C,rot_local)
c      write(damp_out,*) ' Add_damp1_ld ok'
c---------------------------------------★整合質量(ok)
      call Add_damp2_ld_pa(n_member1,n_member2,n_istep,ld_point,Member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
c      write(damp_out,*) ' Add_damp2_ld ok'
c---------------------------------------★部材減衰(ok)
       call Add_damp3_ld_pa(n_member1,n_member2,n_istep,ld_point,Member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ac_member,Newmark_P,Model_type.n_m_damp)
c      write(damp_out,*) ' Add_damp3_ld ok'
c---------------------------------------★線形剛性項計算(ok)
c                                                  レーリー減衰も含む
      call Add_stiff1_ld_pa(n_member1,n_member2,n_istep,ld_point,Member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ak_linear,Newmark_P)
c      write(damp_out,*) ' Add_stiff1_ld ok'
```

```
c————————————————————————————————★Δｔ秒後の変位と速度を予測(ok)
      nx =0
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point,Newmark_P)
c       write(damp_out,*) ' Estimate_disp_vel ok'
c————————————————————————————————★Maxwell 型モデルの計算(ok)
      call Add_fdd_ld_pa(n_member1,n_member2,ld_point,E_model6_real,
     *       Element,Member,est_vel_point,rot_memb)
c       write(damp_out,*) ' Add_fdd_ld ok'
c————————————————————————————————★解析時間のセット
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_LD, dTime_LD)
      All_Time_LD = All_Time_LD + dTime_LD    !  右辺項の足し込み
c———— ★★スレーブより右辺項を受信
c     write(damp_out, *) 'n_proc=', n_proc
      if( n_proc .ge. 2)then
c———— ★★  Time Check Codes
           call MPI_WTime(TIME_NET_LD, nStart)
c————————————————————————————————
      call recv_ld_repeat(n_unknown,ld_point_repeat,
     +                    Ms_table,Ms_free,n_proc,
     *                    buf_disp)
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_NET_LD, dTime_NET_LD)
      All_Time_NET_LD = All_Time_NET_LD + dTime_NET_LD    ! 右辺項の通信
c————————————————————————————————
      endif
c————————————————————————————————★線形方程式を解く(ok)
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_CAL, nStart)
c————————————————
      n_skyline=Parameter_C.n_skyline
      call solve_sky(n_skyline,n_unknown,n_unknown,
     *             max_h_sky,gskym,gskym_d,
     *             nwork,twork,ld_point,result_acc_point)
c     write(damp_out,*) ' solve_sky ok'
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_CAL, dTime_CAL)
      All_Time_CAL = All_Time_CAL + dTime_CAL !  振動方程式を解く
      call MPI_WTime(TIME_DISP_VEL, nStart)
c————————————————————————————————★加速度をスレーブに送信(ok)
c———— ★★スレーブに計算結果を転送
      if( n_proc .ge. 2)then
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_NET_ACC, nStart)
c————————————————————————————————
c           write(damp_out, '(a, i5)') 'Send_Result_acc:istep = ', istep
      call Send_Results_acc(n_unknown,result_acc_point,
     *         nm_parallel,n_proc, Ms_table, Ms_free,
     *         S_control_code,buf_disp)
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_NET_ACC, dTime_NET_ACC)
      All_Time_NET_ACC = All_Time_NET_ACC + dTime_NET_ACC
c————————————————————————————————
      endif
c————————————————————————————————★加速度よりβ法に基づき変位と速度を計算(ok)
      call Cal_disp_vel(n_unknown,
     * result_disp_point, result_vel_point, result_acc_point,
     * past_disp_point, past_vel_point, past_acc_point,
     * Newmark_P)
c       write(damp_out,*) ' Cal_disp_vel ok'
c———— ★★  Time Check Codes
c————————————————————————————————★解析時間のセット
      call MPI_WTime(TIME_DISP_VEL, dTime_DISP_VEL)
      All_Time_DISP_VEL = All_Time_DISP_VEL + dTime_DISP_VEL    ! 変位と加速度の計算,予測
      goto 9980
c————————————————————————————————————————————————————————————
c
c     ★         非線形解析
c
c————————————————————————————————————————————————————————————
 9981 continue
c————————————————————————————————————————————————————————————
c
c     ★         動的解析 (反復解法)
c
c————————————————————————————————————————————————————————————
c———— ★★  Time Check Codes  Newmarkβ法
      call MPI_WTime(TIME_NEWMARK, nStart)          !  Newmarkβ法の反復解法部分
      call MPI_WTime(TIME_LD, nStart)                  !  右辺項の作成
c————————————————————————————————★右辺の定数ベクトルゼロセット(ok)
      call Clear_vec(n_unknown,ld_point )
c           write(damp_out,*) ' Clear_vec ok', Control.jikuzero
c————————————————————————————————★部材節点力のセット(ok)
      call Get_pointforce_ld_pa(n_member1,n_member2,ld_point,Member)
c     write(damp_out, *) 'Get_pointforce_ld_pa ok'
c————————————————————————————————★地震加速度セット(ok)
      acc1=Get_Acc(T,1,acc_earth,Dynamic_load,Newmark_P.f1_T)
      acc2=Get_Acc(T,2,acc_earth,Dynamic_load,Newmark_P.f1_T)
      acc3=Get_Acc(T,3,acc_earth,Dynamic_load,Newmark_P.f1_T)
c       write(damp_out,'(a,4f10.3,i4)') ' Get_Acc ok'
c————————————————————————————————★集中質量に関する慣性項
      call Add_earth1_ld(n_istep,acc1,acc2,acc3,ld_point,
     *           Point,n_point,am_point,rot_local,Parameter_C)
c           write(damp_out,*) ' Add_earth1_ld ok'
```

```
c————————————————————————————★整合質量に関する慣性項
      call Add_earth2_ld(acc1,acc2,acc3,ld_point,
     *    Member,n_member,am_member,rot_local,Parameter_C,Dynamic_load)
c      write(damp_out,*) ' Add_earth2_ld ok'
c————————————————————————————★節点荷重のセット(ok)
      p1=Get_Ps(T,1,fdd_point,Dynamic_load)
      p2=Get_Ps(T,2,fdd_point,Dynamic_load)
      p3=Get_Ps(T,3,fdd_point,Dynamic_load)
c      write(damp_out,'(a,4f10.3)') ' Get_Ps ok'
      call Add_point_ld(p1,p2,p3,ld_point,n_unknown,
     *       Dynamic_load,fld_static)
c      write(damp_out,*) ' Add_point_ld ok'
c————————————————————————————★線形減衰項計算(ok)
c————————————————————————————★集中質量(ok)
      call Add_damp1_ld(n_istep,ld_point,Point,n_point,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       am_point,Newmark_P,Parameter_C,rot_local)
c      write(damp_out,*) ' Add_damp1_ld ok'
c————————————————————————————★整合質量(ok)
      call Add_damp2_ld_pa(n_member1,n_member2,n_istep,ld_point,Member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
c      write(damp_out,*) ' Add_damp2_ld ok'
c————————————————————————————★部材減衰(ok)
      call Add_damp3_ld_pa(n_member1,n_member2,n_istep,ld_point,Member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ac_member,Newmark_P,Model_type.n_m_damp)
c      write(damp_out,*) ' Add_damp3_ld ok',n_istep
c————————————————————————————★線形剛性項計算(ok)
c                                                  レーリー減衰も含む
      call Add_stiff1_ld_pa(n_member1,n_member2,n_istep,ld_point,Member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ak_linear,Newmark_P)
c      write(damp_out,*) ' Add_stiff1_ld ok'
c     write(damp_out, '(a)') 'Add_stiff1_ld_pa'
c     write(damp_out, '(5e16.5)') (ld_point(i), i=1, n_unknown)
c————————————————————————————★Δｔ秒後の変位と速度を予測(ok)
      n_err_roop = 0
 9991 continue
      nx =0
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point,Newmark_P)
c      write(damp_out,*) ' Estimate_disp_vel ok'
c————————————————————————————★解析時間のセット
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_LD, dTime_LD)
      All_Time_LD = All_Time_LD + dTime_LD    !  右辺項の足し込み
c————————————————————————————————————————————————————————
c
c    ★         反復計算開始
c
c————————————————————————————————————————————————————————
      n_roop=Newmark_P.max_repeat
      do iroop=1,n_roop
      write(damp_out,*) ' 反復回数：',iroop
c———— ★★  Time Check Codes 非線形項作成
      call MPI_WTime(TIME_NONLINER, nStart)
c————————————————————————————★反復に関連する右辺ベクトルのゼロセット(ok)
      call Clear_vec(n_unknown,ld_point_repeat)
c           write(damp_out,*) ' Clear_vec ok'
c————————————————————————————★線形剛性に関するベクトル(ok)
      call Add_stiff2_ld_pa(n_member1,n_member2,ld_point_repeat,
     *       Member,n_member,est_disp_point,ak_linear)
c      write(damp_out,*) ' Add_stiff2_ld ok'
c————————————————————————————★接線剛性に関する増分ベクトル(ok)
      call Add_tan_stiff_ld_pa(n_member1,n_member2,ld_point_repeat,
     *       Member,n_member,est_ddisp_point,ak_nonlinear)
c      write(damp_out,*) ' Add_tan_stiff_ld ok'
cc————————————————————————————★Maxwell 型モデルの計算(ok)
      call Add_fdd_ld_pa(n_member1,n_member2,ld_point_repeat,
     *       E_model6_real,Element,Member,est_vel_point,rot_memb)
c      write(damp_out,*) ' Add_fdd_ld ok'
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_NONLINER, dTime_NONLINER)
      All_Time_NONLINER = All_Time_NONLINER + dTime_NONLINER    ! 非線形項の足し込み
c————————————————————————————★右辺項を受け取り、総和をとる（新帯）
      if (n_proc .ge. 2)then
c———— ★★スレーブより右辺項を受信
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_NET_LD, nStart)
c————————————————————————————————————————
c           write(damp_out,*) ' enter recv_ld_repeat'
      call recv_ld_repeat(n_unknown,ld_point_repeat,
     +                           Ms_table, Ms_free,n_proc,
     *                    buf_disp)
c———— ★★  Time Check Codes
      call MPI_WTime(TIME_NET_LD, dTime_NET_LD)
      All_Time_NET_LD  = All_Time_NET_LD + dTime_NET_LD   ! 右辺項通信
c————————————————————————————————————————
      endif
c     write(damp_out,*) ' recv_ld_repeat out'
c————————————————————————————★右辺２項の和を取る(ok)
c———— ★★  Time Check Codes 非線形項作成
      call MPI_WTime(TIME_NONLINER, nStart)
      call add_vec(n_unknown,ld_point_repeat,ld_point)
```

```
c————— ★★  Time Check Codes
      call MPI_WTime(TIME_NONLINER, dTime_NONLINER)
      All_Time_NONLINER = All_Time_NONLINER + dTime_NONLINER   ! 非線形項の足し込み

c——————————————————————————————★線形方程式を解く(ok)
c      write(76,'(a)')' ld_point_repeat'
c      write(76,'(10e12.4)')(ld_point_repeat(i),i=1,n_unknown)
c————— ★★  Time Check Codes 振動方程式を解く
      call MPI_WTime(TIME_CAL, nStart)
      n_skyline=Parameter_C.n_skyline
       call solve_sky(n_skyline,n_unknown,n_unknown,
     *             max_h_sky,gskym,gskym_d,
     *             nwork,twork,ld_point_repeat,result_acc_point)
c      write(76,'(a)')' result_acc_point'
c      write(76,'(10e12.4)')(result_acc_point(i),i=1,n_unknown)
c      write(damp_out, '(a, 5i5)') 'solve_sky ok'
c——————————————————————————★解析時間のセット
c————— ★★  Time Check Codes
      call MPI_WTime(TIME_CAL, dTime_CAL)
      All_Time_CAL = All_Time_CAL + dTime_CAL !  振動方程式を解く
c————— ★★  Time Check Codes
      call MPI_WTime(TIME_DISP_VEL, nStart)
c——————————————————————————————★β法に基づき加速度より変位と速度を計算(ok)
       call Cal_disp_vel(n_unknown,
     * result_disp_point, result_vel_point, result_acc_point,
     * past_disp_point, past_vel_point, past_acc_point,
     * Newmark_P)
      call MPI_WTime(TIME_DISP_VEL, dTime_Time_DISP_VEL)
      All_Time_DISP_VEL = All_Time_DISP_VEL + dTime_Time_DISP_VEL
c——————————————————————————————★収束したかチェック(ok)
       ite_end=lCheck_error(iroop,n_point,Point,n_unknown,
     *         result_disp_point,est_disp_point, Newmark_P)
       if(S_control_code.ne.2) S_control_code = ite_end
c——————————————————————————————★加速度をスレーブに送信(ok)
c————— ★★スレーブに計算結果を転送
       if( n_proc .ge. 2)then
c————— ★★  Time Check Codes 予測加速度の通信
      call MPI_WTime(TIME_NET_ACC, nStart)
c——————————————————————————————
      call Send_Results_acc(n_unknown,result_acc_point,
     *         nm_parallel,n_proc, Ms_table,Ms_free,
     *         S_control_code,buf_disp)
c————— ★★  Time Check Codes 予測加速度の通信
      call MPI_WTime(TIME_NET_ACC, dTime_NET_ACC)
      All_Time_NET_ACC = All_Time_NET_ACC + dTime_NET_ACC ! 予測加速度通信
c——————————————————————————————
      endif
       if(S_control_code .eq. 2) goto 9998                    ! 解発散、後処理へ
       if(S_control_code .eq. 0) goto 9980                    ! 収束、応力計算へ
```

```
c——————————————————————————————★次の増分値を予測(ok)
      nx=iroop
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point, Newmark_P)
c       write(damp_out,*) ' Estimate_disp_vel ok'
      end do
c————————————————————————————————————
c
c    ★          収束しなかった時の後処理
c
c————————————————————————————————————
      write(76,*) '  収束エラー発生、解析終了'
      n_iterate = 9999
      ierr_dat =30
      call err_outf(ieer_dat)
c————————————————————————————————————
c
c    ★          スレーブに終了処理を通知
c
c————————————————————————————————————
      return
 9982 continue

c————————————————————————————————————
c
c    ★          計算終了・後処理開始
c
c————————————————————————————————————
 9980 continue
c     write(damp_out,*) "応力計算"
c     write(damp_out,*) "部材両端の節点力"
c——————————————————————————★解析時間のセット
c————— ★★  Time Check Codes Newmarkβ法
      call MPI_WTime(TIME_NEWMARK, dTime_NEWMARK)
      All_Time_NEWMARK = All_Time_NEWMARK + dTime_NEWMARK !  Newmarkβ法  反復解法
c——————————————————————————————★解析結果・増分変位をセット(ok)
      call Set_ddisp(n_unknown,est_ddisp_point,
     *     result_disp_point,past_disp_point)
c       write(damp_out,*) ' Set_disp_vel_acc ok'
c——————————————————————————————★変位、速度、加速度を出力
      vacc(1)=Get_Acc(T,1,acc_earth,Dynamic_load,Newmark_P.f1_T)
      vacc(2)=Get_Acc(T,2,acc_earth,Dynamic_load,Newmark_P.f1_T)
      vacc(3)=Get_Acc(T,3,acc_earth,Dynamic_load,Newmark_P.f1_T)
c       write(damp_out,'(a,4f10.3)') ' Get_Acc ok'
      call Out_disp_vel_acc(Point,n_point,Parameter_C,
     *     past_disp_point, past_vel_point, past_acc_point,
```

```
     *     rot_local,ifl,iflz,vacc,i_print)
c       write(damp_out,*) ' Out_disp_vel_acc ok'
c      write(6,*) ' Out_disp_vel_acc ok'
c-------------------------------------------------★変位、速度、加速度の最大値セット(ok)
      call Get_max_disp(Max_disp,Parameter_C.n_point,Point,
     *      past_disp_point, past_vel_point, past_acc_point,
     *      d_max_v,id_max_v,vacc)
c       write(damp_out,*) ' Get_max_disp ok'
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_STRESS, nStart)
c-----------------------------------------------------------------------
      call Cal_stress_pa (n_member1,n_member2,
     *     Member,n_member,Model_type,Element,
     *     past_disp_point,past_vel_point,est_ddisp_point,rot_memb,
     *     E_model6_real,ak_nonlinear)
c       write(damp_out,*) ' Cal_stress_pa ok'
c-------------------------------------------------★計算時間のセット
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_STRESS, dTime_STRESS)
      All_Time_STRESS = All_Time_STRESS + dTime_STRESS    ! 応力計算
      call MPI_WTime(TIME_F_STRESS, nStart)   ! ファイバー応力の計算開始
c-----------------------------------------------------------------------
c
c     ★         部材の弾塑性状態をチェック
c
c-----------------------------------------------------------------------
c-------------------------------------------------★部材塑性をチェック
c-------------------------------------------------★部材両端の節点力計算 (ok)
c     write(damp_out,*)"部材塑性をチェック"
c-------------------------------------------------★ファイバー応力セット
      call Check_stress_pa(n_member1,n_member2,Control,
     *     Control.type_analysis,
     *     ak_nonlinear,Member,n_member,Model_type,
     *     Element,past_disp_point,est_ddisp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,
     *     Bilinear_work,Trilinear_work,Concrete_work,RO_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member,
     *     S_comp_model,E_modelx,M_modelx,
     *     E_fiber_work,M_fiber_work)
c           write(damp_out,*) ' Check_stress ok',Newmark_P.max_repeat
c-------------------------------------------------★Maxwell 型モデルの非線形性チェック(ok)
      call Check_Maxwell_stress_pa(n_member1,n_member2,Member,
     *     Element,E_model6_real,Newmark_P)
c      write(damp_out,*) ' Check_Maxwell_stress ok'
c-------- ★★部材両端の節点力を受け取る
      if(i_print.eq.0) then                          ! ファイル出力のときのみデータを取得
      if( n_proc .ge. 2)then
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_NET_FORCE, nStart)
c-----------------------------------------------------------------------
c           write(damp_out, '(a, i5)') 'recv_force_vpp:istep = ', istep
      call recv_force_vpp( Parameter_C,Member,pa_work_force,
     +                                                  nm_parallel,n_proc)
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_NET_FORCE, dTime_NET_FORCE)
      All_Time_NET_FORCE = All_Time_NET_FORCE + dTime_NET_FORCE! 部材端節点力通信
c-----------------------------------------------------------------------
      endif
      endif

c-------------------------------------------------★計算時間のセット
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_F_STRESS, dTime_F_STRESS)
      All_Time_F_STRESS = All_Time_F_STRESS + dTime_F_STRESS    ! ファイバー応力計算
c-------------------------------------------------★部材応力を出力
      call Out_stress(Member,Element,E_model6_real,M_model11,
     *                M_model12,M_model13,M_model15,M_model21,M_model22,
     *                M_model31,M_model32,M_model33,
     *               n_member,ifl,iflz,i_print,Out_section,S_comp_model)
c       write(damp_out,*) ' Out_stress ok'
c-------------------------------------------------★部材応力を出力
      call Out_Fiber(Member,Element,E_model11,M_model11,
     *               E_model12,M_model12,E_model13,M_model13,
     *               E_model15,M_model15,
     *               E_model21,M_model21,E_model22,M_model22,
     *               E_model31,M_model31,E_model32,M_model32,
     *               E_model33,M_model33,
     *               E_model_fiber,M_model_fiber,
     *               n_member,ifl,iflz,i_print,Out_section,
     *               S_comp_model, E_modelx, M_modelx,
     *               E_fiber_work, M_fiber_work)
c       write(damp_out,*) ' Out_stress ok'
c-------------------------------------------------★応力等の最大値セット
      call Get_max_stress_pa(n_member1,n_member2,Member,Max_stress)
c      write(damp_out,*) ' Get_max_stress ok'

c-------------------------------------------------★解析結果・変位、速度、加速度をセット(ok)
      call Set_disp_vel_acc(n_unknown,est_ddisp_point,
```

```
     *     result_disp_point, result_vel_point, result_acc_point,
     *     past_disp_point, past_vel_point,past_acc_point,
     *     past_dacc_point)                          ! past_dacc_point に過去の加速度をセットする
c       write(damp_out,*) ' Set_disp_vel_acc ok'
c-------------------------------------------------★不釣合い力（反力と荷重）の出力(ok)
      call Get_pointforce(fll_force_point,Member,n_member,
     *                    Point,n_point)
c       write(damp_out,*) ' Get_pointforce ok'
      call Out_pointforce(fll_force_point,Point,rot_local,
     *                    n_point,ifl(1),iflz(1),i_print)
c-------------------------------------------------★部材節点力の描画用データセット
      if(i_read_ndbalanceF .ne. 0  .and.
     *  istep .eq. ns_step+n_step - 1 ) then
      call Set_preset_nd(fll_force_point,Point,rot_local,n_point,
     *                   F_ndbalanceF)
      endif
c-------------------------------------------------★不釣合い力の計算
c-------------------------------------------★節点静的荷重項(ok)
      if(ifl(2).eq.1.and.i_print.eq.0) then

      p1=Get_Ps(T,1,fdd_point,Dynamic_load)
      p2=Get_Ps(T,2,fdd_point,Dynamic_load)
      p3=Get_Ps(T,3,fdd_point,Dynamic_load)
      call Add_pointforce(p1,p2,p3,fll_force_point,
     *                    fll_static_point,n_point)
c-------------------------------------------★集中質量に関する地震項
      vacc(1)=Get_Acc(T,1,acc_earth,Dynamic_load,Newmark_P.f1_T)
      vacc(2)=Get_Acc(T,2,acc_earth,Dynamic_load,Newmark_P.f1_T)
      vacc(3)=Get_Acc(T,3,acc_earth,Dynamic_load,Newmark_P.f1_T)
      call Add_earth1_pointforce(n_istep,vacc,fll_force_point,
     *           Point,n_point,am_point,rot_local,Parameter_C)
c-------------------------------------------★整合質量に関する地震項
      call Add_earth2_pointforce(vacc,fll_force_point,
     *    Member,n_member,am_member,rot_local,Parameter_C,Dynamic_load)
c-------------------------------------------------★慣性項と減衰項計算(ok)
c-------------------------------------------★集中質量の慣性項と減衰項(ok)
      call Add_damp1_pointforce(n_istep,fll_force_point,Point,n_point,
     *       past_vel_point,past_acc_point,
     *       am_point,Newmark_P,Parameter_C,rot_local)
c-------------------------------------------★整合質量の慣性項と減衰項(ok)
      call Add_damp2_pointforce(n_istep,fll_force_point,Member,n_member,
     *       past_vel_point,past_acc_point,
     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
c-------------------------------------------★部材減衰の減衰項(ok)
      call Add_damp3_pointforce(n_istep,fll_force_point,Member,n_member,
     *       past_vel_point,ac_member,Newmark_P,Model_type.n_m_damp)
c-------------------------------------------★線形剛性のレーリー減衰項
      call Add_stiff1_pointforce(n_istep,fll_force_point,Member,
     *       n_member,past_vel_point,ak_linear,Newmark_P)
c-------------------------------------------★不釣合い力の出力(ok)
       call Out_pointforce(fll_force_point,Point,rot_local,
     *                     n_point,ifl(2),iflz(2),i_print)

      endif
c-------------------------------------------------★終了か？（ステップと最大値チェック）
      if(istep .ge. Newmark_P.nn_step .or.
     *    d_max_v .gt.Control.collapse_maxdisp ) S_control_code=2
c           write(damp_out,*) ' Check_error ok'
c-------------------------------------------------★接線剛性の計算(ok)
c     write( damp_out,*)"接線剛性の計算"
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_KT, nStart)          ! 接線剛性の計算開始
c-----------------------------------------------
      call Get_nonlinear_stiff_pa(n_member1,n_member2,
     *     Control.type_analysis,ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,S_comp_model,E_modelx, M_modelx,
     *     E_fiber_work, M_fiber_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member )
c-------------------------------------------★解析時間のセット
c-------- ★★  Time Check Codes
      call MPI_WTime(TIME_KT, dTimeKT)
      All_Time_KT = All_Time_KT + dTimeKT      !  接線剛性作成
c-------------------------------------------★解析時間のセット
      call MPI_WTime(TIME_STEP, dTime_STEP)
      All_Time_STEP = All_Time_STEP + dTime_STEP    ! ステップ毎の計算時間
 9999 continue
c-----------------------------------------------------------------------
c
c    ★         時間増分更新
c
c-----------------------------------------------------------------------
      iend_code = 0
      ns_step=ns_step + n_step
c-----------------------------------------------------------------------
c
c    ★         描画用データのセット
c
```

```
c-----------------------------------------------------------------------
c-------------------------------------★変位
      if(i_read_disp .ne. 0) then
c     write(damp_out,*) ' Set_preset_disp ok'
c------- ★★描画用データのセット
      call Set_preset_disp(1,n_point,past_disp_point,F_disp,Point,
     *                          rot_local,Parameter_C)
      endif
c-------------------------------------★応力
      if(i_read_spring .ne. 0) then
      call Set_preset_spring(Member,Element,E_model6_real,
     *               M_model11,M_model12,M_model13,M_model15,
     *               M_model21,M_model22,
     *               n_member,F_fay,F_n_spring,F_my_spring,
     *               F_mz_spring,i_stat_spring,S_comp_model)
c     write(damp_out,*) ' Set_preset_spring ok'
      endif

c------- ★★  Parallel Code - Start  モニターにデータ転送
c     call ContactMonitor(i_calnum,iend_code,icontrol,ierr_dat,
c     *            T,dt,n_step,ns_step,d_max_v,id_max_v,
c     *            i_read_disp,F_disp,i_read_ndbalanceF,F_ndbalanceF,
c     *            i_read_spring,F_fay,F_n_spring,F_my_spring,
c     *            F_mz_spring,i_stat_spring,n_iterate,
c     *            nm_iterate,numb_method)
c
cc-----------------------------------------------------------------------
c
c     ★         ファイルを強制的に書き出し
c
c-----------------------------------------------------------------------
c     call CompulsorilyWrite(ifl,iflz)
c     call COMMITQQ( damp_out)
c-----------------------------------------------------------------------
      return
c-----------------------------------------------------------------------
c
c     ★         応答解析終了処理
c
c-----------------------------------------------------------------------
 9998 continue
c-----------------------------------------------------------------------
c
c     ★         描画用データのセット
c
c-----------------------------------------------------------------------
c-------------------------------------★変位
      if(i_read_disp .ne. 0) then
c------- ★★描画用データのセット
      call Set_preset_disp(1,n_point,past_disp_point,F_disp,Point,
     *                          rot_local,Parameter_C)
c     write(damp_out,*) ' Set_preset_disp ok'
      endif
c-------------------------------------★応力
      if(i_read_spring .ne. 0) then
      call Set_preset_spring(Member,Element,E_model6_real,
     *               M_model11,M_model12,M_model13,M_model15,
     *               M_model21,M_model22,
     *               n_member,F_fay,F_n_spring,F_my_spring,
     *               F_mz_spring,i_stat_spring,S_comp_model)
c     write(damp_out,*) ' Set_preset_spring ok'
      endif
c-------------------------------------★部材節点力の描画用データセット
      if(i_read_ndbalanceF .ne. 0  .and.
     *  istep .eq. ns_step+n_step - 1 ) then
      call Set_preset_nd(fll_force_point,Point,rot_local,n_point,
     *                   F_ndbalanceF)
      endif
c------- ★★  Parallel Code - Start  モニターにデータ転送
      ns_step = -1
c     call ContactMonitor(i_calnum,iend_code,icontrol,ierr_dat,
c     *            T,dt,n_step,ns_step,d_max_v,id_max_v,
c     *            i_read_disp,F_disp,i_read_ndbalanceF,F_ndbalanceF,
c     *            i_read_spring,F_fay,F_n_spring,F_my_spring,
c     *            F_mz_spring,i_stat_spring,n_iterate,
c     *            nm_iterate,numb_method)
 9997 continue
c-------------------------------------★最大変位、速度、加速度の出力
      call Out_max_disp(Max_disp,Parameter_C.n_point,
     *                  Point,ifl(12),iflz(12))
c     write(damp_out,*) ' Out_max_disp ok'
c------- ★★  最大応力等をスレーブからの受信
      if( n_proc .ge. 2)then
            call recv_max_stress( Parameter_C,Max_stress,
     +            nm_parallel,n_proc)
      endif
c-------------------------------------★最大応力等の出力
      call Out_max_stress(Max_stress,Parameter_C.n_member,
     *                    ifl(16),iflz(16))
c     write(damp_out,*) ' Out_max_stress ok'
c-------------------------------------★解析時間のセット
c------- ★★  Time Check Codes
c     call MPI_WTime(TIME_STEP, dTime_STEP)
c     All_Time_STEP = All_Time_STEP + dTime_STEP    ! ステップ毎の計算時間
      call MPI_WTime(TIME_ALL, All_Time_ALL)              ! 総解析時間
c-------------------------------------★解析時間の出力
      write(76,'(a)')
     +                '解析時間の出力        :      秒'
```

```
      write(76,'(a,e16.5)')
     +                '予備計算              :',All_Time_PRE / 1000000
      write(76,'(a,e16.5)')
     +                '係数行列作成          :',All_Time_K / 1000000
      write(76,'(a,e16.5)')
     +                'Newmarkβ法            :',All_Time_NEWMARK / 1000000
      write(76,'(a,e16.5)')
     +                '-線形右辺項作成       :',All_Time_LD / 1000000
      write(76,'(a,e16.5)')
     +                '-非線形項作成         :',All_Time_NONLINER / 1000000
      write(76,'(a,e16.5)')
     +                '--通信時間(非線形項)  :',All_Time_NET_LD / 1000000
      write(76,'(a,e16.5)')
     +                '-振動方程式を解く     :',All_Time_CAL / 1000000
      write(76,'(a,e16.5)')
     +                '--通信時間(予測加速度):',All_Time_NET_ACC / 1000000
      write(76,'(a,e16.5)')
     +                '-変位、速度の計算     :',All_Time_DISP_VEL / 1000000
c     write(76,'(a,e16.5)')
c     +                '--通信時間(制御コード):',All_Time_NET_CONV / 1000000
      write(76,'(a,e16.5)')
     +                '応力の計算            :',All_Time_STRESS / 1000000
      write(76,'(a,e16.5)')
     +                '--通信時間(応力)      :',All_Time_NET_FORCE / 1000000
      write(76,'(a,e16.5)')
     +                'ファイバー応力の計算  :',All_Time_F_STRESS / 1000000
      write(76,'(a,e16.5)')
     +                '接線剛性作成          :',All_Time_KT / 1000000
      write(76,'(a,e16.5)')
     +                '総解析時間            :',All_Time_STEP / 1000000
      write(76,'(a,e16.5)')
     +                '総時間                :',All_Time_ALL / 1000000
c-------------------------------------★ファイルのクローズ
      do i=1,16
      if(ifl(i).eq.1) close(iflz(i))
      enddo
      write(damp_out,*) ' ファイルのクローズ ok'
c-------------------------------------★ファイルのタイムスタンプ
      ihan = 0
      NFILE=100
      call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      call fltime(ifl,ifly,N_analysis,iflout)
      write(damp_out,*) ' ファイルのタイムスタンプ ok',FNX_file
      ihan = 1
      if(iflout.eq.1) call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      write(damp_out,*) ' データセット ok'
c-----------------------------------------------------------------
c
c     ★         動的配列の解放
c
c-----------------------------------------------------------------
c-----------------★         配列の動的領域を解放する（その１）
      if(M_alloc(1).eq.1)then
      DEALLOCATE (Point,Element,Member,Max_disp,Max_stress)        !ok
      DEALLOCATE (fll_static_point,am_point,fll_force_point)       !ok
      DEALLOCATE (am_member,rot_memb_t,rot_memb,ak_linear,ak_nonlinear)       !ok
      N=Parameter_C.n_local_coord
      if(N.ne.0) DEALLOCATE (rot_local)       !ok
      N = Parameter_C.n_S_comp_model
      if(N.ne.0) DEALLOCATE (S_comp_model)
      N = Parameter_C.nE_New_Element
      if(N.ne.0) DEALLOCATE (E_Fiber_work)
      endif
c-----------------★         配列の動的領域を解放する（その２）
      if(   M_alloc(2).eq.1)then
      N=Model_type.n_m_damp
      if(N.ne.0) DEALLOCATE (ac_member)       !ok
      n= Model_type.n_e_New_fiber
      if(N.ne.0) DEALLOCATE (E_modelx)
      n= Model_type.n_m_New_fiber
      if(N.ne.0) DEALLOCATE (M_modelx)
      n=Parameter_C.nM_New_Element
      if(N.ne.0) DEALLOCATE (M_Fiber_work)

      endif
c-----------------★         配列の動的領域を解放する（その４）
      if(   M_alloc(4).eq.1)then
c      DEALLOCATE (test_vector)                          !テスト用
      DEALLOCATE (disp_point,vel_point,acc_point,
     *     est_disp_point,est_vel_point,est_ddisp_point)
      DEALLOCATE (result_disp_point,result_vel_point,result_acc_point,
     *   past_disp_point,past_vel_point,past_acc_point,
     *   past_dacc_point)
      DEALLOCATE (ld_point,ld_point_repeat,fld_static)
      DEALLOCATE (a_vector,b_vector)
      DEALLOCATE (max_h_sky)
      if(n_proc.ge.2) DEALLOCATE (Ms_table,Ms_free)
      endif
c-----------------★         配列の動的領域を解放する（その５）
      if( M_alloc(5).eq.1)then
      DEALLOCATE (gskym,gskym_d,nwork,twork)         !ok
      endif
cc       DEALLOCATE (work1_elememt,work2_elememt,work1_member,work2_member)
c-----------------★         配列の動的領域を解放する（その２）
      if(   M_alloc(2).eq.1)then
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      DEALLOCATE (RO_work )
```

```
      endif
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      DEALLOCATE (MSS_work )            !ok
      endif
      n=Model_type.n_m_model(6)
      if(n.ne.0) then
      DEALLOCATE (E_model6_real )           !ok
      endif
      n= Model_type.n_e_model(11)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model11 )               !ok
      endif
      n= Model_type.n_m_model(11)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model11 )              !ok
      endif
      n= Model_type.n_e_model(12)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model12 )              !ok
      endif
      n= Model_type.n_m_model(12)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model12 )              !ok
      endif
      n= Model_type.n_e_model(18)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model13 )              !ok
      endif
      n= Model_type.n_m_model(18)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model13 )              !ok
      endif
      n= Model_type.n_e_model(15)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model15 )              !ok
      endif
      n= Model_type.n_m_model(15)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model15 )              !ok
      endif
      n= Model_type.n_e_model(13)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model21 )              !ok
      endif
      n= Model_type.n_m_model(13)        !部材モデルの数
      if(n.ne.0) then
       DEALLOCATE ( M_model21 )              !ok
      endif
      n= Model_type.n_e_model(14)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model22 )               !ok
      endif
      n= Model_type.n_m_model(14)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model22 )              !ok
      endif
      n= Model_type.n_e_model(16)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model31)               !ok
      endif
      n= Model_type.n_m_model(16)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model31 )              !ok
      endif
      n= Model_type.n_e_model(17)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model32)               !ok
      endif
      n= Model_type.n_m_model(17)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model32 )              !ok
      endif
      n= Model_type.n_e_model(19)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model33)               !ok
      endif
      n= Model_type.n_m_model(19)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model33 )              !ok
      endif
      if(Dynamic_load.n_load_dynamic .ne. 0) then
      DEALLOCATE (acc_earth)               !ok
      endif
      if(Dynamic_load.n_load_point .ne. 0) then
      DEALLOCATE (fdd_point)               !ok
      endif
      endif
c-----------★        配列の動的領域を解放する（その３）
      if( M_alloc(3).eq.1)then
      n= Model_type.nm_div_fmodel         !要素モデル内のサブ要素の数
      if(n.ne.0) then
      DEALLOCATE (M_model_fiber )              !ok
      endif
      n= Model_type.nm_div_felement        !ファイバー要素のエレメント最大数
      if(n.ne.0) then
      DEALLOCATE (E_model_fiber )              !ok
      endif
```

```
      endif
c-----------------★        配列の動的領域を解放する（その５）
      if( M_alloc(5).eq.1)then
      n= Model_type.n_m_bilinear          !  バイリニアの履歴要の数
      if(n.ne.0) then
      DEALLOCATE (Bilinear_work )              !ok
      endif
      n= Model_type.n_m_trilinear          !  トリリニアの履歴要の数
      if(n.ne.0) then
      DEALLOCATE (Trilinear_work )              !ok
      endif
      n= Model_type.n_m_Concrete          !  コンクリートの履歴要の数
      if(n.ne.0) then
      DEALLOCATE (Concrete_work )              !ok
      endif
      endif
c-----------------------------------------------------------------------
c
c     ★        並列処理で使用した動的配列の解放
c
c-----------------------------------------------------------------------
      if(n_proc.ge.2) then
      DEALLOCATE (nm_parallel)
      DEALLOCATE (pa_work_force)
      DEALLOCATE (buf_disp)
      endif
c-------------------------------------------★計算終了コードセット
      iend_code = 1
      close (damp_out)
      return
c-----------------------------------------------------------------------
c
c     ★        解析終了
c
c-----------------------------------------------------------------------
      end
```

付 1.2 スレーブ主サブルーチン

```
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C     ●   SUBROUTINE /submain_dynamic_S
C      ------------------------------
C         Parallel version for Slave
C      ------------------------------
C     ●   動的解析スレーブ用主プログラム（反復解法）Ver.1.11
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
c     スレーブ並列処理用　メインプログラム
c      基本計画
c         部材に関する動的領域は担当部材のみ確保する。
c         要素及び節点データに関しては全てマスターと同じとする
c         自由度に関するデータは担当部材が必要とする領域のみ使用する
c         必要な情報は、マスターがファイルより取得し、スレーブには転送する
c
c      ** 解析番号と担当部材をマスターから取得
c
c        １．予備計算　：　部材に関するデータのみ処理する
c        ２．反復計算　：　圧縮した加速度を受信する
c                          圧縮した右辺項を送信する
c        ３．応力計算　：　マスターが出力するための応力を送信する
c
c
c
c スレーブ側がマスターに送受信するデータブロックは以下のようである。
c    Ⅰ。予備計算
c         1. コントロール情報受信（全スレーブ同じ情報）recv_ctlset( )
c         2. 構造データ受信（全スレーブ同じ情報）recv_structure( )
c         3. 初期不整データ受信（全スレーブ同じ情報）注１recv_imperfection( )
c         4. ファイバーデータ受信（全スレーブ同じ情報）注１recv_Fiber( )
c         5. RO モデル用データ受信（全スレーブ同じ情報）注１recv_RO_data( )
c         6. 減衰用データ受信（全スレーブ同じ情報）注１recv_damp( )
c         7. 初期変位受信（全スレーブ同じ情報：現在使用不可）recv_init_disp( )
c         8. 初期応力受信（全スレーブ同じ情報：現在使用不可）recv_init_stress( )
c            注１担当部材にこの部材モデルがなくても、解析モデルで使用している場合は
c                送信されるため受信なければならない。
c    Ⅱ。動的解析
c         1. 右辺項を送信 send_ld_repeat( )
c         2. 計算結果加速度データ受信 recv_result_acc( )
c         3. 未収束・収束コード受信（全スレーブ同じ情報）bcast_ite( )
c         4. 部材両端の節点力を送信 send_force_vpp( )
c    Ⅲ。後処理
c         1. 最大応力等をスレーブからの送信 send_max_stress( )
c
c スレーブでは、送受信数が異なる。マスターから送信されたデータは、そのまま使用する。
c 各スレーブでは、担当部材に従って、未知数番号を作り直す。
c
c         1. スレーブ用テーブル作成　Reset_ij_table( )注２
c            注２：節点変位を同一視する場合、その相手が担当部材に連結していない場合
c                  を考慮しなければならない。
c
c
c-----------------------------------------------------------------------
c
c       Model_No.1 = 1              !  幾何学非線形型有限要素弾性モデル
c       Model_No.2 = 2              !  3 次元せん断弾塑性モデル
```

```
c        Model_No.3 = 3              !  3 次元軸力弾塑性モデル
c        Model_No.4 = 4              !  3 次元ケーブル弾塑性モデル
c        Model_No.5 = 5              !  3 次元免振モデル（MSS モデル）
c        Model_No.6 = 6              !  3 次元制震 Maxwell モデル
c        Model_No.7 = 7              !  3 次元弾塑性バネモデル(*)
c
c
c        Model_No.11 = 11            !  両端ファイバーモデル
c        Model_No.12 = 12            !  両端、中央ファイバーモデル
c        Model_No.13 = 18            !  両端ピンで中央ファイバーモデル
c        Model_No.15 = 15            !  ファイバーモデル
c
c        Model_No.13 = 21            !  両端 MS モデル
c        Model_No.14 = 22            !  両端、中央 MS モデル
c
c        Model_No.31 = 31            !  両端アナロジーモデル
c        Model_No.32 = 32            !  両端、中央アナロジーモデル
c        Model_No.33 = 19            !  両端ピン、中央アナロジーモデル
c
c        Model_No.51 = 51            !  3 次元プレテンション動作モデル
c
c        Model_No.50 = 1001          !  DLL 有限要素弾塑性モデル
c
c-------------------------------------------------------------------------
c
c      ファイバー履歴タイプ
c      nm_type: 1      バイリニア型
c        No.1 = 1                    !  対称バイリニア
c        No.2 = 2                    !  対称トリリニア
c        No.3 = 3                    !  直線コンクリート型
c        No.4 = 4                    !  曲線線コンクリート型
c        No.5 = 5                    !  対称バイリニア型(移動＋等方効果用)
c        No.6 = 6                    !  対称トリリニア型（移動＋等方効果用）
c        No.7 = 7                    !  非対称バイリニア型
c        No.8 = 8                    !  非対称トリリニア型（移動＋等方効果用）
c
c        アナロジーモデル履歴タイプ
c              11      完全弾塑性型
c              12      等方硬化弾塑性型
c              13      移動硬化弾塑性型
c              14      等方硬化＋移動硬化弾塑性型
c
c
c        マルチスプリング履歴タイプ
c              21      武田モデル
c              22      等方硬化＋移動硬化トリリニア型
c
c
c-------------------------------------------------------------------------
c
c      断面モデル
c        No.1                        !  円管
c        No.2                        !  角パイプ
c        No.3                        !  H　型
c        No.4                        !  長方形
c        No.5                        !  T　型
c        No.6                        !
c        No.7                        !
c
c-------------------------------------------------------------------------
c
c    部材モデルの階層構造は、以下のようである。
c    部材ー＞要素モデルー＞断面モデルー＞履歴モデル
c    １．部材は、各部材固有のデータを保持する。
c    ２．要素モデルは、部材の内部を構成するデータを保持する。
c    ３．断面モデルは、部材の断面形状を等のデータを保持する。
c    ４．履歴モデルは、各要素の弾塑性状態を制御するデータを保持する。
c
c
c
c    部材モデルの組み込み手順
c    １．Member_s と Element_s 構造体を設定する。
c    ２．弾塑性解析を実行するために必要となる領域（構造体）を設定する。
c    ３．Cal_stiff_linear に線形剛性を組み込む
c    ４．Cal_stress に応力計算を組み込む
c    ５．Check_stress に弾塑性チェックを組み込む
c    ６．Get_nonlinear_stiff に非線形剛性を組み込む
c
c
c
c
c     解析種別　N_analysis：
c          7: 線形解析
c          8: 幾何学的非線形解析
c          9: 弾塑性解析
c         10: 幾何学的＋弾塑性解析
c          6: 線形座屈解析
c
c    部材別解析種別：Member(i).nm_analysis
c         -1: 線形解析
c          0: 全体解析種別に従う
c          1: 部材座標系 x 方向弾性
c          2: 部材座標系 y 方向弾性
c          3: 部材座標系 z 方向弾性
c
c-------------------------------------------------------------------------
c
c submain_dynamic_a　引数
```

```
c              iend_code       :計算終了コード 0=継続  1:終了
c              icontrol        :計算コード     0:予備計算  1:動的解析 99:終了処理
c              ierr_dat        :エラーコード
c              T               :計算時刻
c              dt              :解析増分時間
c              n_step          :今回の解析ステップ数
c              ns_step         :解析開始ステップ（最初はゼロセットが必要）
c              n_iterate       :反復回数
c-------------------------------------------------------------------------
c-------------------------------------------------------------------------
c
c    ★         サブルーチン定義
c
c-------------------------------------------------------------------------
c-------------------------------------------------------------------------
      subroutine submain_dynamic_S(i_calnum,iend_code,icontrol,ierr_dat)
c-------------------------------------------------------------------------
c-----------------------------------------------★並列処理に必要な宣言
      use MPI_DEFINE
      implicit real*8(A-H,O-Z)
c-----------------------------------------------★計算結果のダンプ出力ファイル番号
      parameter(damp_out = 76)
      character*20 :: strDampFile       ! DOUTPUT_S○のファイル名
c-----------------------------------------------★構造体の定義ヘッダーファイル
      include "..¥..¥sf3st¥submain.h"
      include "..¥..¥sf3st¥submainx.h"
      include "..¥..¥sf3st¥New_submain.h"
c-------------------------------------------------------------------------
c
c    ★         静的配列定義
c
c-------------------------------------------------------------------------
      dimension ifl(16),ifly(16),idfile(100),iflz(16)
      character fnx_file*100,TITLEX*50
      dimension fs_st(10,3),fl_st(10,3),ifp_st(10,3),ist(3)
      dimension igra(3),xgal(3),hh(100),qhh(100),vacc(6)
      dimension No_section(10)
      dimension M_alloc(10)                          ! 動的配列確保のチェック
      dimension dt_M_filter(4)    ! Maxwell  フィルターデータ
c-------------------------------------------------------------------------
c
c    ★     解析手法のコントロールデータ
c
c-------------------------------------------------------------------------
      integer N_implicit_method    !陰解法の回数  －１：無限回
      integer Iteration_method     !反復計算：１、(陰解法：２:使用せず)
      integer S_control_code      ! スレーブコントロールコード
c-------------------------------------------------------------------------
c
```

```
c    ★         動的配列定義
c
c-------------------------------------------------------------------------
c-------------------------------------------------------------------------
c      real*8  gskym(n_skyline)        ! スカイライン行列
c-------------------------------------------------------------------------
c     １）データ取得バッファ
      real*8, save, ALLOCATABLE  :: pa_work_force(:,:)
      real*4, ALLOCATABLE :: ff_data(:)
      integer, ALLOCATABLE ::if_data(:),iif_dt(:,:),ibuf(:)
      real*8, ALLOCATABLE :: buf(:)
c     ２）自由度関連
      real*8, ALLOCATABLE ::
     *   disp_point(:),vel_point(:),acc_point(:),
     *   est_disp_point(:),est_vel_point(:),est_ddisp_point(:)
      real*8, ALLOCATABLE ::
     *   result_disp_point(:),result_vel_point(:),result_acc_point(:),
     *   past_disp_point(:),past_vel_point(:),past_acc_point(:),
     *   past_dacc_point(:)
      real*8, ALLOCATABLE ::
     *   ld_point(:),ld_point_repeat(:)
c     *   ,a_vector(:),b_vector(:),fld_static(:,:)
c      integer, ALLOCATABLE :: max_h_sky(:)
c-------------------------------------------------------------------------
c      real*8  disp_point(n_unknown)    ! 節点変位
c      real*8  vel_point(n_unknown)     ! 節点速度
c      real*8  acc_point(n_unknown)     ! 節点加速度

c      real*8  est_disp_point(n_unknown)    ! 予測節点変位
c      real*8  est_vel_point(n_unknown)     ! 予測節点速度
c      real*8  est_ddisp_point(n_unknown)   ! 予測節点増分変位

c      real*8  result_disp_point(n_unknown)    ! 計算結果節点変位
c      real*8  result_vel_point(n_unknown)     ! 計算結果節点速度
c      real*8  result_acc_point(n_unknown)     ! 計算結果節点加速度
c      real*8  past_dacc_point(n_unknown)      ! 計算結果節点増分加速度

c      real*8  past_disp_point(n_unknown)    ! Δt 秒前の節点変位
c      real*8  past_vel_point(n_unknown)     ! Δt 秒前の節点速度
c      real*8  past_acc_point(n_unknown)     ! Δt 秒前の節点加速度

c      real*8  ld_point(n_unknown)            ! 線形右辺項ベクトル
c      real*8  ld_point_repeat(n_unknown)     ! 反復用右辺項ベクトル
c
c      real*8  a_vecter(n_unknown)            ! 動的解析用 work ベクトル
c      real*8  b_vector(n_unknown)            ! 動的解析用 work ベクトル
c
c      real*8  fld_static(3,n_unknown)        !  節点荷重の分布（釣合式に用いた座標系）
c      real*8  fld_dynamic(3,n_unknown)       !  地震に対する荷重ベクトル
```

```
c      integer  max_h_sky(n_unknown+1)       !  スカイライン行列の各列の高さ
c-------------------------------------------------------------------
c-------------------------------------------------------------------
c     ４） 節点
c      real*8, ALLOCATABLE :: fll_static_point(:,:,:),
c     *                       am_point(:,:),fll_force_point(:,:)
c-------------------------------------------------------------------
c      real*8  fll_static_point(3,6,n_point) !  節点荷重の分布（全体座標系）
c      real*8  fll_force_point(3,n_point)    !  節点不釣合い力の分布（全体座標系）
c      real*8  am_point(2,n_point)           !  節点集中質量
c-------------------------------------------------------------------
c     ５） 部材
      real*8, ALLOCATABLE :: am_member(:,:,:),ac_member(:,:,:)
c-------------------------------------------------------------------
c      real*8  am_member(12,12,n_member)     ! 質量整合行列（釣合座標系）
c      real*8  ac_member(12,12,nc_member)    ! 線形部材減衰行列（釣合座標系）
c-------------------------------------------------------------------
c      ６） 座標変換行列
      real*8, ALLOCATABLE ::rot_memb_t(:,:,:),rot_memb(:,:,:,:),
     *                      rot_local(:,:,:)
c-------------------------------------------------------------------
c      real*8  rot_memb_t(3,3,n_member)        !  部材の全体座標系への回転行列
c      real*8  rot_memb(3,3,2,n_member)        !  部材両端の釣合座標系への回転行列
c      real*8  rot_local(3,3, n_local_coord)   !  全体座標系から局所座標への回転行列
c-------------------------------------------------------------------
c      ７）剛性行列等
      real*8, ALLOCATABLE ::ak_linear(:,:,:),ak_nonlinear(:,:,:)
c-------------------------------------------------------------------
c       real*8  ak_linear(12,12,n_member)       !  部材の線形剛性行列（釣合座標系）
c       real*8  ak_nonlinear(12,12,n_member)    !  部材の接線剛性行列（部材座標系）
c-------------------------------------------------------------------
c      ８）  部材の非線型状態を表す。ワークエリア
      real*8, ALLOCATABLE ::work1_elememt(:),work2_elememt(:),
     *                      work1_member(:),work2_member(:)
c-------------------------------------------------------------------
c      real*8, ALLOCATABLE ::test_vector(:)
c      real*8   work1_elememt(n_request1* n_element_dll)
c      real*8   work2_elememt(n_request2* n_element_dll)
c      real*8   work1_member(n_request3* n_member_dll)
c      real*8   work2_member(n_request4* n_member_dll)
c      注：上記のワークエリアは DLL 内で structure を定義して使用する。
c-------------------------------------------------------------------
c
c    ★       新構造体定義
c
c-------------------------------------------------------------------
      record / S_comp_model_s / S_comp_model
      record / E_modelx_s     / E_modelx
      record / M_modelx_s     / M_modelx
      record / M_model_mss_s  / M_model_mss
      record / M_model_ang_s  / M_model_ang
      record / E_Fiber_Work_s / E_Fiber_Work
      record / M_Fiber_Work_s / M_Fiber_Work
c-------------------------------------------------------------------
c
c    ★       構造体定義
c
c-------------------------------------------------------------------
      record /control_s      / Control
      record /dynamic_load_s / Dynamic_load
      record /parameter_s    / Parameter_C
      record /n_model_s      / Model_type
      record /newmark_s      / Newmark_P
      record /out_section_s  / Out_section
      record /point_s        / Point
      record /element_s      / Element
      record /member_s       / Member
      record /max_stress_s   / Max_stress
c-------------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c
c-------------------------------------------★Model_No.2 3 次元せん断弾塑性モデル
      record / RO_work_s /  RO_work
c-------------------------------------------★Model_No.3 3 次元軸力弾塑性モデル
c      record /element3_s / Element
c      record /member3_s  / Member
c-------------------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
c      record /element4_s / Element
c      record /member4_s  / Member
c-------------------------------------------★Model_No.5 3 次元 MSS 免振モデル
      record / MSS_work_s   / MSS_work
c-------------------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      record / E_model6_real_s   / E_model6_real
c-------------------------------------------★Model_No.7 3 次元バネモデル
c      record /element7_s / Element
c      record /member7_s  / Member
c      record / E_model7_real_s    / E_model7_real
c-------------------------------------------★Model_No.11 両端ファイバーモデル
      record / E_model11_s    / E_model11
      record / M_model11_s    / M_model11
c-------------------------------------------★Model_No.12 両端、中央ファイバーモデル
      record / E_model12_s    / E_model12
      record / M_model12_s    / M_model12
c-------------------------------------------★Model_No.13 両端ピン、中央ファイバーモ
デル
      record / E_model13_s    / E_model13
      record / M_model13_s    / M_model13
c-------------------------------------------★Model_No.15 両端ファイバーFEM モデル
```

```
      record / E_model15_s    / E_model15
      record / M_model15_s    / M_model15
c-----------------------------------------★Model_No.21 両端 MS モデル
      record / E_model21_s    / E_model21
      record / M_model21_s    / M_model21
c-----------------------------------------★Model_No.22 両端、中央 MS モデル
      record / E_model22_s    / E_model22
      record / M_model22_s    / M_model22
c-----------------------------------------★Model_No.31 両端アナロジーモデル
      record / E_model31_s    / E_model31
      record / M_model31_s    / M_model31
c-----------------------------------------★Model_No.32 両端、中央アナロジーモデル
      record / E_model32_s    / E_model32
      record / M_model32_s    / M_model32
c-----------------------------------------★Model_No.33 両端ピン、中央アナロジーモ
デル
      record / E_model33_s    / E_model33
      record / M_model33_s    / M_model33
c-----------------------------------------★Model_No.51 3 次元プレテンション動作モ
デル
c      record / E_model51_s    / E_model51
c      record / M_model51_s    / M_model51
c-----------------------------------------★ファイバーモデル用エリア
      record / E_model_fiber_s   / E_model_fiber
      record / M_model_fiber_s   / M_model_fiber
c-----------------------------------------★履歴モデル用エリア
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
c-------------------------------------------------------------------
c
c    ★         新構造体の動的領域宣言
c
c-------------------------------------------------------------------
      ALLOCATABLE :: S_comp_model(:)
      ALLOCATABLE :: E_modelx(:)
      ALLOCATABLE :: M_modelx(:)
      ALLOCATABLE :: M_model_mss(:)
      ALLOCATABLE :: M_model_ang(:)
      ALLOCATABLE :: E_Fiber_Work(:)
      ALLOCATABLE :: M_Fiber_Work(:)
c-------------------------------------------------------------------
c
c    ★         構造体の動的領域宣言
c
c-------------------------------------------------------------------
      ALLOCATABLE :: Bilinear_work(:)
      ALLOCATABLE :: Trilinear_work(:)
      ALLOCATABLE :: Concrete_work(:)
      ALLOCATABLE :: Point(:)
      ALLOCATABLE :: Element(:)
      ALLOCATABLE :: Member(:)
      ALLOCATABLE :: Max_stress(:)
c-----------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c-----------------------------------------★Model_No.2 3 次元せん断弾塑性モデル
      ALLOCATABLE :: RO_work(:)
c-----------------------------------------★Model_No.3 3 次元軸力弾塑性モデル
c-----------------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
c-----------------------------------------★Model_No.5 3 次元免振モデル
      ALLOCATABLE :: MSS_work(:)
c-----------------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      ALLOCATABLE :: E_model6_real(:)
c-----------------------------------------★Model_No.7 3 次元バネモデル
c       ALLOCATABLE ::  E_model7(:)
c-----------------------------------------★Model_No.11 両端ファイバーモデル
      ALLOCATABLE :: E_model11(:)
      ALLOCATABLE :: M_model11(:)
c-----------------------------------------★Model_No.12 両端、中央ファイバーモデル
      ALLOCATABLE :: E_model12(:)
      ALLOCATABLE :: M_model12(:)
c-----------------------------------------★Model_No.12 両端、中央ファイバーモデル
      ALLOCATABLE :: E_model13(:)
      ALLOCATABLE :: M_model13(:)
c-----------------------------------------★Model_No.15 両端ファイバーFEM モデル
      ALLOCATABLE :: E_model15(:)
      ALLOCATABLE :: M_model15(:)
c-----------------------------------------★Model_No.21 両端 MS モデル
      ALLOCATABLE :: E_model21(:)
      ALLOCATABLE :: M_model21(:)
c-----------------------------------------★Model_No.22 両端、中央 MS モデル
      ALLOCATABLE :: E_model22(:)
      ALLOCATABLE :: M_model22(:)
c-----------------------------------------★Model_No.31 両端アナロジーモデル
      ALLOCATABLE :: E_model31(:)
      ALLOCATABLE :: M_model31(:)
c-----------------------------------------★Model_No.32 両端、中央アナロジーモデル
      ALLOCATABLE :: E_model32(:)
      ALLOCATABLE :: M_model32(:)
c-----------------------------------------★Model_No.33 ピン 両端､中央アナロジーモ
デル
      ALLOCATABLE :: E_model33(:)
      ALLOCATABLE :: M_model33(:)
c-----------------------------------------★Model_No.51 3 次元プレテンション動作モ
デル
c       ALLOCATABLE :: E_model51(:)
c       ALLOCATABLE :: M_model51(:)
c-----------------------------------------★ファイバーモデル用エリア
      ALLOCATABLE :: E_model_fiber(:)
```

```
      ALLOCATABLE :: M_model_fiber(:)
c-----------------------------------------------------------------------
c
c     ★         新構造体の静的確保
c
c-----------------------------------------------------------------------
      save S_comp_model
      save E_modelx
      save M_modelx
      save M_model_mss
      save M_model_ang
      save E_Fiber_Work
      save M_Fiber_Work
c-----------------------------------------------------------------------
c
c     ★         重要データの静的確保
c
c-----------------------------------------------------------------------
      save ifl ,ifly ,idfile ,iflz
      save fnx_file ,TITLEX ,M_alloc
      save Parameter_C,Control,Dynamic_load,Model_type,Newmark_P
      save Point,Element,Member,Max_stress
      save disp_point,vel_point,acc_point,
     *     est_disp_point,est_vel_point,est_ddisp_point
      save result_disp_point,result_vel_point,result_acc_point,
     *     past_disp_point,past_vel_point,past_acc_point,
     *     past_dacc_point
      save ld_point,ld_point_repeat
c     save a_vector,b_vector,fld_static
c      save fll_static_point,  fdd_point,am_point,fll_force_point
      save am_member,ac_member
      save rot_memb_t,rot_memb,rot_local
      save ak_linear,ak_nonlinear
      save work1_elememt,work2_elememt,work1_member,work2_member

      save Bilinear_work,Trilinear_work,Concrete_work
      save Out_section,n_istep

c     save test_vector
c-----------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c-----------------------------------------★Model_No.2 3次元せん断弾塑性モデル
      save RO_work
c-----------------------------------------★Model_No.3 3次元軸力弾塑性モデル
c-----------------------------------------★Model_No.4 3次元ケーブル弾塑性モデル
c-----------------------------------------★Model_No.5 3次元免振モデル
      save MSS_work
c-----------------------------------------★Model_No.6 3次元制震 Maxwell モデル
      save E_model6_real
c-----------------------------------------★Model_No.7 3次元バネモデル
c     save E_model7
c-----------------------------------------★Model_No.11 両端ファイバーモデル
      save   E_model11
      save   M_model11
c-----------------------------------------★Model_No.12 両端、中央ファイバーモデル
      save   E_model12
      save   M_model12
c-----------------------------------------★Model_No.13 両端ピン、中央ファイバーモ
デル
      save   E_model13
      save   M_model13
c-----------------------------------------★Model_No.15 両端ファイバーFEM モデル
      save   E_model15
      save   M_model15
c-----------------------------------------★Model_No.21 両端 MS モデル
      save   E_model21
      save   M_model21
c-----------------------------------------★Model_No.22 両端、中央 MS モデル
      save   E_model22
      save   M_model22
c-----------------------------------------★Model_No.31 両端アナロジーモデル
      save   E_model31
      save   M_model31
c-----------------------------------------★Model_No.32 両端、中央アナロジーモデル
      save   E_model32
      save   M_model32
c-----------------------------------------★Model_No.33 両端ピン、中央アナロジーモ
デル
      save   E_model33
      save   M_model33
c-----------------------------------------★Model_No.51 3次元プレテンション動作モ
デル
c     save    E_model51
c     save    M_model51
c-----------------------------------------★ファイバーモデル用エリア
      save    E_model_fiber
      save    M_model_fiber
c-----------------------------------------★解析手法
      save N_implicit_method
      save Iteration_method,stimes
      save S_control_code
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
c     ★         システムからの制御情報を取得（以後実行文）
c
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c       if(icontrol .eq. 1  ) goto 9990    ! 解析処理へ
```

```
c       if(icontrol .eq. 99 ) goto 9997    ! 終了処理へ
c―――――― ★★解析制御情報をマスターから取得
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★並列処理用予備計算（処理）（前）
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★MPI 上での自己の状態を得る
c―――――― ★★
c     自分のランクを取得
      call MPI_Comm_rank(MPI_COMM_WORLD, n_myRank, nErr)
c     プロセスの総数を取得
      call MPI_Comm_size(MPI_COMM_WORLD, n_proc, nErr)
      write(*, '(A, i3, A, i3)'), "myRank=", n_myRank, "n_proc=", n_proc
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Doutput の出力先変更
      write(strDampFile, '(A10,i3,A5)') 'DOUTPUT_S',n_myRank,'.txt'
      open (damp_out,FILE = strDampFile)
      write(damp_out, '(A10, i3)') 'rank: ', n_myRank
c―――――― ★★担当部材の設定
      call GetRange(1, n_myRank, n_member1)   ! 担当部材の先頭番号
      call GetRange(2, n_myRank, n_member2)   ! 担当部材の最終番号
      max_member=n_member2-n_member1+1       ! 担当部材の最大数

      ALLOCATE (pa_work_force(33,max_member))
      ALLOCATE (ff_data(400),if_data(400),iif_dt(2,4))

      do i=1, 400
            ff_data(i) = i
            if_data(i) = i
      end do
c     write(damp_out,'(a,i5)')'n_proc=',n_proc
      write(*,'(/a,i5,a,i5,a,i5)')'max_member=',max_member,
     +      ' n_member1=',n_member1,' n_member2=',n_member2
      write(damp_out,'(a,i5,a,i5,a,i5)')'max_member=',max_member,
     +      ' n_member1=',n_member1,' n_member2=',n_member2
c      担当部材をモニターより取得
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★モニターからのコントロール情報を取得
(ok)
c       call sysnam_pa(FNX_file,N_analysis)
      do i=1,10
      M_alloc(i)=0                        ! 動的配列の確保をチェックする配列をゼロセット
      enddo
      ihan = 0
      ierr = 0
      NFILE=100
      ierr_dat =0
c―――――― ★★解析制御情報をマスターから取得
      write(*, *) "enter recv_ctlset  xxxxxxxxx"
c―――――― ★★解析制御情報をマスターから取得(vpp)
      call recv_ctlset(Parameter_C,ff_data,if_data,iif_dt,N_analysis,
     *                 MPP_Ana_Group,ierr_dat)
      write(*, *) "out recv_ctlset : Parameter_C.n_element =",
     *            Parameter_C.n_element
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★制御情報の取得に失敗した場合はここで戻る
      if(ierr_dat .ne. 0 ) then
      DEALLOCATE (ff_data,if_data,iif_dt)
      goto 9997
      endif
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★動的解析ダイアログその１のデータをセ
ット(ok)
      is=iif_dt(1,1)                      !  ff_data() 先頭番地
      js=iif_dt(2,1)                      !  if_data() 先頭番地
      call dyctl1_set(ierr,NINDIT,GINDIS,F1SEC,fs_st,fl_st,ifp_st,IST,
     *              JIKUZERO,G_JIKUZERO_ALPH,ff_data(is),if_data(js))
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★動的解析ダイアログその２のデータをセ
ット(ok)
      is=iif_dt(1,2)
      js=iif_dt(2,2)
      call dyctl2_set(ierr,NSTEP,F2SEC,DELT,IGRA,IBETA,BETA,GUMMA,XGAL,
     *            NNTIME,EPSDSP,load_memb_mass,dt_M_filter,IT_ANALYS,
     +               ff_data(is),if_data(js))
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★解析結果の出力パラメータをセット(ok)
      is=iif_dt(1,3)
      js=iif_dt(2,3)
      call doutcl_set(ierr,IWSTP,SOUTSC,DMAXCK,No_section,
     +               ff_data(is),if_data(js))
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――★減衰ダイアログのデータをセット(ok)
      is=iif_dt(1,4)
      js=iif_dt(2,4)
      call damctl_set(ierr,NREAD,ITYDP,NDMP,NDMP2,NHH,HH,QHH,
     +               ff_data(is),if_data(js))
      DEALLOCATE (ff_data,if_data,iif_dt)
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★         制御情報を構造体にセット
c
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――

      call Set_control(Control,N_analysis,NINDIT,GINDIS,
     *                 IWSTP,SOUTSC,DMAXCK,
     *                 in_disp,in_stres,IT_ANALYS,
     *                  JIKUZERO,G_JIKUZERO_ALPH)        !ok (in_disp,in_stres  未定義)
      write(*,*) ' Set_control Ok'
      call Set_model_type(Model_type)
      write(*,*) ' Set_model_type Ok'
      call Set_newmark(Newmark_P,F1SEC,F2SEC,HH(1),HH(2),
     *                 QHH(1),QHH(2),DELT,BETA,EPSDSP,NNTIME,GUMMA,
     *                 ITYDP,NDMP,NDMP2)
      write(*,*) ' Set_newmark Ok'
      call Set_dynamic_load(Dynamic_load,IST,IGRA,
     *                      XGAL,load_memb_mass)
      write(*,*) ' Set_dynamic_load Ok'
```

```
c-----------------------------------------------------------------------
c
c     ★   DLL を使用する場合、ワークエリアとして必要な大きさ及び DLL 番号を得る
c                                       DLL subroutine code No. 1
c
c-----------------------------------------------------------------------
c       if(Parameter_C.n_element_dll .ne. 0 )
c     * call Get_DLL_code(dll_code,n_request1,n_request2,
c     *                   n_request3,n_request4)
c-----------------------------------------------------------------------
c
c     ★         構造体の大きさを動的確保する（その１）
c
c-----------------------------------------------------------------------
      ALLOCATE (Member(max_member))           ! 担当部材数
      ALLOCATE (Element(Parameter_C.n_element))
      ALLOCATE (Point(Parameter_C.n_point))
      ALLOCATE (Max_stress(Parameter_C.n_member))
c-----------------------------------------------------------------------
c
c     ★         新配列の大きさを動的確保する
c
c-----------------------------------------------------------------------
      N= Parameter_C.n_S_comp_model       ! 任意静的縮合型モデル
      write(*,*) ' Parameter_C.n_S_comp_model ',n

      if(N.ne.0) then

      N = Parameter_C.nE_New_Element      !  任意型静的縮合モデルに含まれる要素エレメント数
      write(*,*) ' Parameter_C.nE_New_Element ',n
      if(N.ne.0)then
      ALLOCATE (E_Fiber_work(N))          ! 新規縮合モデルの要素エレメント数の動的確保  ①-2
      endif
      write(*,*) ' Parameter_C.nE_New_Element ',n
c     write(76,*) ' Parameter_C.nE_New_Element ',n
      nx=1
      call recv_S_comp_model(nx,nx_file,S_comp_model,Model_type ,ieerx,
     *                       id_buf,jd_buf,buf,ibuf)
      Parameter_C.n_S_comp_model=nx_file
c     write(*,*) ' Parameter_C.n_S_comp_model 1',id_buf,jd_buf,nx_file
c     write(76,*) ' Parameter_C.n_S_comp_model 1',id_buf,jd_buf,nx_file
      ALLOCATE (S_comp_model(nx_file))          ! 新規縮合モデル設定領域の動的確保
      ALLOCATE (buf(id_buf+10),ibuf(jd_buf+10))
      nx=2
c     write(*,*) ' Parameter_C.n_S_comp_model 2',nx_file
      call recv_S_comp_model(nx,nx_file,S_comp_model,Model_type ,ieerx,
     *                       id_buf,jd_buf,buf,ibuf)
      DEALLOCATE (buf,ibuf)
c-----------------------------------------------------------------------
c
c     ★         E_Fiber_work の内容を別途送信
c
c-----------------------------------------------------------------------
      call recv_E_fiber(E_Fiber_work )
      endif
c-----------------------------------------------------------------------
c
c     ★         配列の大きさを動的確保する
c
c-----------------------------------------------------------------------
      N=max_member              ! 担当部材数
      ALLOCATE (
     *   am_member(12,12,N),
     *   rot_memb_t(3,3,N),rot_memb(3,3,2,N),
     *   ak_linear(12,12,N),ak_nonlinear(12,12,N)
     *         )
      N=Parameter_C.n_local_coord       ! 局所座標系を使用する場合
      if(N.ne.0)ALLOCATE (
     *   rot_local(3,3,N)
     *)
      M_alloc(1)=1
c-----------------------------------------------------------------------
c
c     ★         構造・荷重データをマスターから取得する
c
c-----------------------------------------------------------------------
c-----------------------------------------------★基本構造データをマスターから取得(vpp)
c--------- ★★マスターから構造データを取得
c     バッファデータ仕様
c     buf()     １：座標 3
c               ２：局所座標 3
c               ３：要素 17
c               ４：部材 4
c     ibuf()    １：境界拘束条件 7
c               ２：要素 6
c               ３：部材  14
c
      id_buf=Parameter_C.n_point*6+Parameter_C.n_element*17+
     *       Parameter_C.n_member*4
      jd_buf=Parameter_C.n_point*7++Parameter_C.n_element*6+
     *       Parameter_C.n_member*14                  ! 全部材数用意する
      ALLOCATE (buf(id_buf+10),ibuf(jd_buf+10))
c     write(*, *) ' allocate ',id_buf, jd_buf
c--------- ★★構造データ受信
      call recv_structure(buf,ibuf,id_buf,jd_buf)
c     write(*, *) ' recev_structure ok'
      call set_structure(Point,Member,Element,Parameter_C,
```

```
     *       Model_type,ierr,buf,ibuf,n_member1,n_member2,
     *       S_comp_model,E_Fiber_work)
c     write(*, *) ' set structure ok',max_member
      DEALLOCATE (buf,ibuf)
c     write(76, *) 'set_structure out::Parameter_C.n_element=',
c     *                           Parameter_C.n_element
      Parameter_C.n_member = max_member     !スレーブ側で計算する担当部材数に変換
c-------------------------------★情報の取得に失敗した場合はここで戻る
      if(ierr .ne. 0 ) goto 9997
c-----------------------------------------------------------------------
c
c     ★         制震セミアクティブダンパー用フィルターのパラメータセット
c
c-----------------------------------------------------------------------
      if(Model_type.n_m_model(6) .ne. 0) then
      call set_Maxwell_filter(dt_M_filter,Newmark_P.dt,Model_type,ierr)
      if(ierr.ne.0)     goto 9997
c     write(*,*) ' Set_Maxwell_filter Ok'
      endif
c-----------------------------------------------------------------------
c
c     ★         計算用構造体の大きさを動的確保する（その２）
c
c-----------------------------------------------------------------------
      N=Model_type.n_m_damp
      if(N.ne.0)then
      ALLOCATE (ac_member(12,12,N))
      endif
c-----------------------------------------------★Model_No.1 通常の有限要素弾塑性モデル
c-----------------------------------------------★Model_No.2 3 次元せん断弾塑性モデル
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      ALLOCATE (RO_work(n))
      endif
c-----------------------------------------------★Model_No.3 3 次元軸力弾塑性モデル
c-----------------------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
c-----------------------------------------------★Model_No.5 3 次元免振モデル
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      ALLOCATE (MSS_work(n))
      endif
c-----------------------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      n=Model_type.n_m_model(6)
      if(n.ne.0) then
      ALLOCATE (E_model6_real(n))
      endif
c-----------------------------------------------★Model_No.7 3 次元バネモデル
c      n=Model_type.n_m_model(7)
c     if(n.ne.0) then
c       ALLOCATE (E_model7(n))
c      endif
c-----------------------------------------------★ファイバーモデル
c-----------------------------------------------★Model_No.11 両端ファイバーモデル
      n= Model_type.n_e_model(11)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model11(n))
      endif
      n= Model_type.n_m_model(11)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model11(n))
      endif
c-----------------------------------------------★Model_No.12 両端、中央ファイバーモデル
      n= Model_type.n_e_model(12)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model12(n))
      endif
      n= Model_type.n_m_model(12)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model12(n))
      endif
c-----------------------------------------------★Model_No.13 両端、中央ファイバーモデル
      n= Model_type.n_e_model(18)         !要素モデルの数
      if(n.ne.0) then
       ALLOCATE (E_model13(n))
      endif
      n= Model_type.n_m_model(18)         !部材モデルの数
      if(n.ne.0) then
       ALLOCATE ( M_model13(n))
      endif
c-----------------------------------------------★Model_No.15 両端ファイバーFEM モデル
      n= Model_type.n_e_model(15)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model15(n))
      endif
      n= Model_type.n_m_model(15)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model15(n))
      endif
c-----------------------------------------------★Model_No.21 両端 MS モデル
      n= Model_type.n_e_model(13)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model21(n))
      endif
      n= Model_type.n_m_model(13)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model21(n))
      endif
c-----------------------------------------------★Model_No.22 両端、中央 MS モデル
```

```
      n= Model_type.n_e_model(14)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model22(n))
      endif
      n= Model_type.n_m_model(14)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model22(n))
      endif
c-----------------------------------------★Model_No.31 両端アナロジーモデル
      n= Model_type.n_e_model(16)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model31(n))
      endif
      n= Model_type.n_m_model(16)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model31(n))
      endif
c     write(damp_out,*) ' allocate model 31 ',n
c-----------------------------------------★Model_No.32 両端、中央アナロジーモデル
      n= Model_type.n_e_model(17)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model32(n))
      endif
      n= Model_type.n_m_model(17)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model32(n))
      endif
c     write(damp_out,*) ' allocate model 32 ',n
c-----------------------------------------★Model_No.33 両端ピン、中央アナロジーモ
デル
      n= Model_type.n_e_model(19)         !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model33(n))
      endif
      n= Model_type.n_m_model(19)         !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model33(n))
      endif
c-----------------------------------------★新規モデル Model_No.51-70
      n= Model_type.n_e_New_fiber         !要素モデルの数
c     write(76,'(a,i4)') 'Model_type.n_e_New_fiber',n
      if(n.ne.0) then
      ALLOCATE (E_modelx(n))                                          ! ②-2
      Endif
      n= Model_type.n_m_New_fiber         ! 部材モデルの数
c     write(76,'(a,i4)') 'Model_type.n_m_New_fiber',n
      if(n.ne.0) then
      ALLOCATE (M_modelx(n))                                          ! ④-2
      endif
      n=Parameter_C.nM_New_Element        ! 数新規モデルにおける部材エレメント数
c     write(76,'(a,i4)') 'Parameter_C.nM_New_Element',n
      if(n.ne.0) then
      ALLOCATE (M_Fiber_work(n))                                      ! ⑤-2
      endif
c-----------------------------------------★Model_No.51 3 次元プレテンション動作モ
デル
c      n=Model_type.n_m_model(51)
c     if(n.ne.0) then
c      ALLOCATE (M_model51(n),
c     *          E_model51(n))
c      endif
c-----------------------------------------------------------------------
c
c     ★          配列の大きさを動的確保する（その２：DLL 用）
c
c-----------------------------------------------------------------------
c     if(Parameter_C.n_element_dll .ne. 0 ) then
c      N1=n_request1* Parameter_C.n_element_dll
c      N2=n_request2* Parameter_C.n_element_dll
c      N3=n_request3* Parameter_C.n_member_dll
c      N4=n_request4* Parameter_C.n_member_dll
c      ALLOCATE (
c     *   work1_elememt(N1),work2_elememt(N2),
c     *   work1_member(N3),work2_member(N4)
c     *         )
c     endif
c-----------------------------------------★初期不整データをマスターから取得(ok)
c------- ★★マスターから初期不整データを取得
      if(Control.init_imperfection .ne. 0) then
      n_inperfection = 4*Parameter_C.n_point +1      !    初期不整の節点が不明のため節点使用
      ALLOCATE (buf(n_inperfection+10))
c     write(*,*) ' recv_imperfection in',n_inperfection
      call recv_imperfection(Point,buf, n_inperfection)
c     write(*,*) ' recv_imperfection out',Element(1).element_type
      DEALLOCATE (buf)
c-----------------------------★情報の取得に失敗した場合はここで戻る
c     if(ierr .ne. 0 ) goto 9997
      endif
c-----------------------------------------★システム内要素データをマスターから取
得
c     ★          ファイバーデータ
c-------------------------------------★ファイバーデータの予備データをマスターから取得
(ok)
      n_element=Model_type.n_e_model(11)+Model_type.n_e_model(12)+
     *          Model_type.n_e_model(15)+
     *          Model_type.n_e_model(14)+Model_type.n_e_model(13)+
     *          Model_type.n_e_model(16)+Model_type.n_e_model(17)+
     *          Model_type.n_e_model(18)++Model_type.n_e_model(19)+
```

```
     *          Model_type.n_e_New_fiber
      if(n_element.ne.0) then
c——————— ★★マスターからファイバー用制御データを取得
c     write(*,*) ' recv_Fiber_P 1',n_element,Element(1).element_type
      call recv_Fiber_P(id_buf,jd_buf)
c——————— ★★スレーブにファイバー用バッファをゼロセット
c      write(76,'(a,2i4)') ' buff area ',id_buf,jd_buf
      ALLOCATE (buf(id_buf+10),ibuf(jd_buf+10))
c     write(76,*) ' recv_Fiber_P 2',n_element,id_buf,jd_buf
      call recv_Fiber(0,ierr,Parameter_C.n_member,
     *      Parameter_C.n_element,Member,Element,Model_type,
     *      E_model_fiber,M_model_fiber,E_model11,M_model11,E_model12,
     *      M_model12,E_model13,M_model13,E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,E_model33,M_model33,
     *      M_Fiber_work,E_Fiber_work,E_modelx,M_modelx,Parameter_C,
     *       id_buf,jd_buf,buf,ibuf)

c————————————————————————★情報の取得に失敗した場合はここで戻る
c     if(ierr .ne. 0 ) goto 9997
c     write(*,*) ' Fiber_input ok',Element(1).element_type
c————————————————————————————————★ファイバーモデル領域セット
      n= Model_type.nm_div_fmodel          !要素モデル内のサブ要素の数
      if(n.ne.0) then
      ALLOCATE (M_model_fiber(n))
      endif
      n= Model_type.nm_div_felement         !ファイバー要素のエレメント最大数
      if(n.ne.0) then
      ALLOCATE (E_model_fiber(n))
      endif
      M_alloc(3)=1
c————————————————————————————————★ファイバーモデルデータをマスターから取得
c——————— ★★マスターからファイバー用データを取得
c     write(*,*) ' Fiber_input in',Element(1).element_type
      call recv_Fiber(1,ierr,Parameter_C.n_member,
     *      Parameter_C.n_element,Member,Element,Model_type,
     *      E_model_fiber,M_model_fiber,E_model11,M_model11,E_model12,
     *      M_model12,E_model13,M_model13,E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,E_model33,M_model33,
     *      M_Fiber_work,E_Fiber_work,E_modelx,M_modelx,Parameter_C,
     *       id_buf,jd_buf,buf,ibuf)
      DEALLOCATE (buf,ibuf)
c     write(*,*) ' Fiber_input out'
c—————————————————————————————————————————★制御情報の取得に失敗した場合はここで
戻る
c     if(ierr .ne. 0 ) goto 9997

      endif
```

```
c————————————————————————————————————★修正 RO モデル領域をマスターから取得
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
c——————— ★★マスターから修正 RO モデル用データを取得
      call recv_RO_data(0,n,RO_work,Element,Parameter_C.n_element,
     *                                             ierr,id_buf,jd_buf,buf,ibuf)
c     if(ierr .ne. 0 ) goto 9997
      ALLOCATE (buf(id_buf+10))
      call recv_RO_data(1,n,RO_work,Element,Parameter_C.n_element,
     *                                         ierr,id_buf,jd_buf,buf,ibuf)
      DEALLOCATE (buf)
c————————————————————————★情報の取得に失敗した場合はここで戻る
c     if(ierr .ne. 0 ) goto 9997
c     write(damp_out,*) ' RO_data_input ok'
      endif
c————————————————————————————————————————————★DLL 内要素データを入力
c                                       DLL subroutine code No. 2
c      n_element=Parameter_C.n_member_dll
c      if(n_element .ne. 0) then
c      nfix=5
c     nfi=60
c      call infile(nfi,nfix,ierr)
c     if(ierr.ne.0) then
c      ierr_dat =13
c     return
c     endif
c      call Get_element_dll(work1_elememt,work2_elememt,Parameter_C)
c     close(nfix)
c     endif

c————————————————————————————————————————————★レーリー減衰をマスターから取得(ok)
c——————— ★★マスターから減衰データを取得
c     write(*,*) ' recv_damp in'
      call recv_damp(Newmark_P,ierr)
c     write(damp_out,*) ' recv_damp out'
c————————————————————————★情報の取得に失敗した場合はここで戻る
c     if(ierr .ne. 0 ) goto 9997
c     write(damp_out,*) ' Get_damp Ok'
c———————————————————————————————————————————————————————————
c
c    ★          動的解析のための予備計算を行う
c
c———————————————————————————————————————————————————————————
c————————————————————————————————————————————★部材長さ計算(ok)
      call Cal_member_length(Member,Point,Parameter_C)
c     write(damp_out,*) ' Cal_member_length Ok'
c————————————————————————————————————————————★モデルの初期設定
      call Set_initial_data(Element,Member,Parameter_C,Newmark_P,
     *            E_model6_real,Model_type)
```

```
c     write(*,*) ' Set_initial_data Ok'
c-------------------------------------------------★DLL 非線形エリアの設定
c       if(Parameter_C.n_member_dll .ne. 0) then
c       call Set_nonlinear_S(Element,Parameter_C,
c     *     work1_elememt,work2_elememt,work1_member,work2_member)
c       endif
c-------------------------------------------------★座標変換行列計算
      call Get_rotate_all(rot_memb_t,Parameter_C,Point,Member)
c               write(damp_out,*) ' Get_rotate_all Ok'
c-------------------------------------------------★局所座標変換行列計算
      call Get_rot_local(rot_local,Parameter_C,Point)
c               write(damp_out,*) ' Get_rot_local Ok'
c-------------------------------------------------★釣合座標系標変換行列計算
      call Get_rotate(rot_memb,rot_memb_t,rot_local,
     *                Parameter_C,Member)
c               write(*,*) ' Get_rotate Ok'
c-------------------------------------------------★節点拘束表の作成
c                                                   未知数等をセット
      call Set_restraint_point(Parameter_C,Point,Control)
c     write(76,*) ' Set_restraint_point Ok',Parameter_C.n_unknown
c-------------------------------------------------★部材両端の拘束表作成
c-------- ★★節点拘束表の変更（スレーブ）
      call Reset_ij_table(Parameter_C,Member,Point,
     *      Parameter_C.n_unknown)
c       write(76,*) ' Set_restraint_member Ok',Parameter_C.n_unknown
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
c     ★         配列の大きさを動的確保する（その３）
c
c-----------------------------------------------------------------------
      N=Parameter_C.n_unknown
      ALLOCATE (
     *     disp_point(N),vel_point(N),acc_point(N),
     *     est_disp_point(N),est_vel_point(N),
     *     est_ddisp_point(N)
     *         )
      ALLOCATE (
     *   result_disp_point(N),result_vel_point(N),result_acc_point(N+1),
     *   past_disp_point(N),past_vel_point(N),past_acc_point(N),
     *   past_dacc_point(N)
     *         )
      ALLOCATE (
     *   ld_point(N),ld_point_repeat(N)
c     *   ,a_vector(N),b_vector(N),fld_static(3,N)
     *         )
      M_alloc(4)=1
c-----------------------------------------------------------------------
c-------------------------------------------------★ファイバー用履歴要素
      n= Model_type.n_m_bilinear          !  バイリニアの履歴要の数
      if(n.ne.0) then
      ALLOCATE (Bilinear_work(n))
      endif
      n= Model_type.n_m_trilinear          !  トリリニアの履歴要の数
      if(n.ne.0) then
      ALLOCATE (Trilinear_work(n))
      endif
      n= Model_type.n_m_Concrete          !  コンクリートの履歴要の数
      if(n.ne.0) then
      ALLOCATE (Concrete_work(n))
      endif
      M_alloc(5)=1
c-------------------------------------------------★部材両端中央応力、力のゼロセット(ok)
      call  Set_pointforce_zero(Member,Parameter_C.n_member)
c     write(76,*) ' Set_pointforce_zero Ok'
c-------------------------------------------------★部材応力のゼロセット(ok)
      call  Set_stress_zero(Member,Parameter_C.n_member)
c     write(76,*) ' Set_stress_zero Ok'
c-------------------------------------------------★MSS モデルの初期設定
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      call Cal_MSS_dat(Member,Element,Model_type,
     *                 MSS_work,Parameter_C)
c     write(76,*) ' Cal_MSS_dat Ok'
      endif
c-----------------------------------------------------★平面問題における部材の拘束方向チェック(ok)
      call Check_R_direction(Control.analysis_3D,Parameter_C,
     *                       Member,rot_memb)
c     write(76,*) ' Check_R_direction ok'
c-------------------------------------------------★部材の線形剛性計算(ok)
      call Cal_stiff_linear(Model_type,Element,Member,Parameter_C,
     *      ak_linear,E_model11,E_model_fiber,M_model11,M_model_fiber,
     *      E_model12,M_model12,E_model13,M_model13,E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,
     *      E_model33,M_model33,
     *      Bilinear_work,Trilinear_work,Concrete_work,
     *      work1_elememt,work2_elememt,work1_member,work2_member,
     *      S_comp_model, E_modelx, M_modelx,
     *      E_fiber_work, M_fiber_work)
c     write(76,*) ' Cal_stiff_linear Ok'
c-------------------------------------------------★剛性の釣合座標系への変換(ok)
      call Rotate_stiffness(Parameter_C,ak_linear,rot_memb)
c     write(*,*) ' Rotate_stiffness Ok'
c-------------------------------------------------★部材の減衰行列計算(ok)
      if(Parameter_C.nc_member .ne. 0) then
      call Cal_damp_linear(Element,Member,Parameter_C,ac_member,
```

```
     *            E_model6_real,work1_elememt,
     *            work2_elememt,work1_member,work2_member)
c     write(76,*) ' Cal_damp_linear Ok'
c-------------------------------------------------★部材減衰行列の釣合座標系への変換(ok)
      call Rotate_damp(Parameter_C.n_member,ac_member,rot_memb,Member)
c     write(76,*) ' Rotate_damp Ok'
      end if
c-------------------------------------------------★接線剛性のコピー(ok)
      call Initset_nonlin_stiff(ak_linear,ak_nonlinear,
     *                          Parameter_C.n_member)
c     write(76,*) ' Initset_nonlin_stiff Ok'
c-------------------------------------------------★ベクトルのゼロセット
c-----------------------------------------------★増分前の変位、速度、加速度(ok)
      call Set_zero_v(past_disp_point, past_vel_point,
     *       past_acc_point,past_dacc_point,Parameter_C.n_unknown)
c     write(76,*) ' Set_zero_v Ok'
c-----------------------------------------★最大、最小応力等のゼロセット(ok)
      call Clear_max_stress(Max_stress,Parameter_C.n_member)
c     write(76,*) ' Clear_max_stress Ok'
c-----------------------------------------------------------------------
c
c    ★         静的解析結果等を取り込む
c               (初期変位、初期応力を入力)
c
c-----------------------------------------------------------------------
c-------------------------------------------------★初期変位を入力
      if(Control.init_disp.ne.0) then
c------- ★★マスターから初期変位を取得
c       call recv_init_disp(Point,Member,Parameter_C,ierr)
c-------------------------------★情報の取得に失敗した場合はここで戻る
c     if(ierr .ne. 0 ) goto 9997
c     write(damp_out,*) ' Get_damp Ok'
      endif
c-------------------------------------------------★初期応力を入力
      if(Control.init_stress.ne.0) then
c------- ★★マスターから初期応力を取得
c      call recv_init_stress(Point,Member,Parameter_C,ierr)
c-------------------------------★情報の取得に失敗した場合はここで戻る
c     if(ierr .ne. 0 ) goto 9997
c     write(damp_out,*) ' Get_init_stress Ok'
       endif
C-------------------------------------------------★出力指定した断面がファイバー要素
c-------------------------------------------------★かどうかチェック(ok)
c      call out_section_check_pa(Member,Element,
c     *               Parameter_C.n_member,No_section,Out_section,
c     *               ifl,iflz,i_print)
c-----------------------------------------------------------------------
c
c    ★         解析ステップをセットする
c
c-----------------------------------------------------------------------
      ns_step=1
      n_step=10
      d_max_v=0.
      id_max_v=0
      stimes=secnds(0.0)                        ! 解析時間を計る
c-------------------------------------------------★解析手法のセット
      N_implicit_method = 1  !陰解法は１回で反復法に戻る
      Iteration_method  = 1  !最初は反復法を使用
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
c    ★         予備計算はここで終了 (GUI に戻る)
c
c-----------------------------------------------------------------------

c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
c    ★         動的解析開始
c
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
      i_print = 0
 9990 continue
      n_istep=1
      n_iterate = 0                             !  反復回数ゼロセット
      n_member=Parameter_C.n_member
      n_point =Parameter_C.n_point
      n_unknown=Parameter_C.n_unknown
      n_local_coord=Parameter_C.n_local_coord
      write(*,'(a,3i4)') ' Dynamic analysis start :',
     *                 ns_step,n_step,Dynamic_load.load_dynamic(1)
c       do 9999 istep=ns_step,ns_step+n_step - 1
      do 9999 istep=ns_step,Newmark_P.nn_step       ! 最後までループを抜けない
c------- ★★解析開始コードを受信(mpp)
c           スレーブ制御コード 0:  解析開始  正常終了の場合は最後は制御コードなし
c                              1:  発散・途中終了
c                              2:  エラーは発生、途中終了
c                              3:  未収束
c           call synchronous_process( MPP_Ana_Group)
c           call recv_ite(ite_end)
c           if(ite_end.ne.0)go to 9998                    ! 解析終了していればループから抜ける
      T=Newmark_P.dt*istep
c     call synchronous_process(MPP_Ana_group)
      i_print=mod(istep,Control.interval_out)
      stimes =secnds(stimes)
c         write(*,'(/a,i6,a,f10.3,a,f12.4)')
```

```
c     *     ' Step No,:',istep,
c     *     '     Time :',T,
c     *     'sec.     times (s):',stimes
c       write(damp_out,'(/a,i6,a,f10.3,a,f12.4)')
c     *     ' Step No,:',istep,
c     *     '     Time :',T,
c     *     'sec.     times (s):',stimes
      stimes=secnds(0.0)
      if(istep.eq.Newmark_P.n2_step) n_istep=2
c-----------------------------------------------------------------------
c
c     ★         動的解析の開始
c
c-----------------------------------------------------------------------
      if(Control.type_analysis .ne. 7) goto 9981
c-----------------------------------------------------------------------
c
c     ★         線形解析
c
c-----------------------------------------------------------------------
c-------------------------------------------★右辺の定数ベクトルゼロセット(ok)
      call Clear_vec(n_unknown,ld_point )
c         write(damp_out,*) ' Clear_vec ok'
c-------------------------------------------★線形減衰項計算(ok)
c-----------------------------------★整合質量(ok)
      call Add_damp2_ld(n_istep,ld_point,Member,n_member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
c       write(damp_out,*) ' Add_damp2_ld ok'
c-----------------------------------★部材減衰(ok)
       call Add_damp3_ld(n_istep,ld_point,Member,n_member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ac_member,Newmark_P,Model_type.n_m_damp)
c     write(damp_out,*) ' Add_damp3_ld ok'
c-------------------------------------------★線形剛性項計算(ok)
c                                              レーリー減衰も含む
      call Add_stiff1_ld(n_istep,ld_point,Member,n_member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ak_linear,Newmark_P)
c     write(damp_out,*) ' Add_stiff1_ld ok'
c-------------------------------------------★Δｔ秒後の変位と速度を予測(ok)
      nx =0
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point,Newmark_P)
c     write(damp_out,*) ' Estimate_disp_vel ok'
c-------------------------------------------★Maxwell 型モデルの計算(ok)
      call Add_fdd_ld(ld_point,E_model6_real,Element,
     *       Member,n_member,est_vel_point,rot_memb)
c     write(damp_out,*) ' Add_fdd_ld ok'
c-------------------------------------------★右辺項を送る (vpp)
c---------- ★★マスターに右辺項を送信
c     write(76, '(a, i5)') 'Send_ld_repeat:istep = ', istep
      call send_ld_repeat(n_unknown,ld_point_repeat)
c-----------------------------------------------★サーバーで計算した加速度を受け取る
 (vpp)
c---------- ★★マスターから計算結果加速度を取得
c     write(76, '(a, i5)') 'recv_result_acc:istep = ', istep
      call recv_result_acc(n_unknown,result_acc_point,ierr,
     *                     S_control_code)
c-----------------------------------------------★加速度よりβ法に基づき変位と速度を計
算(ok)
      call Cal_disp_vel(n_unknown,
     * result_disp_point, result_vel_point, result_acc_point,
     * past_disp_point, past_vel_point, past_acc_point,
     * Newmark_P)
c     write(damp_out,*) ' Cal_disp_vel ok'
      goto 9980
c-----------------------------------------------------------------------
c
c     ★         非線形解析
c
c-----------------------------------------------------------------------
 9981 continue
c-----------------------------------------------------------------------
c
c     ★         動的解析 (反復解法)
c
c-----------------------------------------------------------------------
c-------------------------------------------★右辺の定数ベクトルゼロセット(ok)
      call Clear_vec(n_unknown,ld_point )
c         write(damp_out,*) ' Clear_vec ok'
c-------------------------------------------★部材節点力のセット(ok)
c     do i=1, n_member
c          write(76, '(a,i4, 10f16.5)')
c     +          'Member::force', i, (Member(i).force(j), j=1,12)
c     end do
       call Get_pointforce_ld(ld_point,Member,n_member)
c     write(76,'(a,10f16.5)')
c     +          'GetPointforce_ld::ld_point', (ld_point(j),j=1,n_unknown)
c-------------------------------------------★線形減衰項計算(ok)
c-----------------------------------★整合質量(ok)
      call Add_damp2_ld(n_istep,ld_point,Member,n_member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
c       write(damp_out,*) ' Add_damp2_ld ok'
```

```
c     write(76,'(a,10f16.5)')
c     +          'Add_damp2_ld::ld_point', (ld_point(j),j=1,n_unknown)
c-------------------------------------★部材減衰(ok)
      call Add_damp3_ld(n_istep,ld_point,Member,n_member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ac_member,Newmark_P,Model_type.n_m_damp)
c       write(damp_out,*) ' Add_damp3_ld ok',n_istep
c     write(76,'(a,10f16.5)')
c     +          'Add_damp3_ld::ld_point', (ld_point(j),j=1,n_unknown)
c-------------------------------------★線形剛性項計算(ok)
c                                        レーリー減衰も含む
      call Add_stiff1_ld(n_istep,ld_point,Member,n_member,
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ak_linear,Newmark_P)
c       write(damp_out,*) ' Add_stiff1_ld ok'
c     write(76,'(a,10f16.5)')
c     +          'Add_stiff1_ld::ld_point', (ld_point(j),j=1,n_unknown)
c-------------------------------------★Δｔ秒後の変位と速度を予測(ok)
C      n_err_roop = 0
 9991 continue
      nx =0
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point,Newmark_P)
c     write(damp_out,*) ' Estimate_disp_vel ok'
c-----------------------------------------------------------------
c
c     ★          反復計算開始
c
c-----------------------------------------------------------------
      n_roop=Newmark_P.max_repeat
      do iroop=1,n_roop
           write(damp_out,*) ' 反復回数：',iroop
c-------------------------------------★反復に関連する右辺ベクトルのゼロセッ
ト(ok)
      call Clear_vec(n_unknown,ld_point_repeat)
c          write(damp_out,*) ' Clear_vec ok'
c-------------------------------------★線形剛性に関するベクトル(vpp)
      call Add_stiff2_ld(ld_point_repeat,
     *       Member,n_member,est_disp_point,ak_linear)
c          write(damp_out,*) ' Add_stiff2_ld_pa ok'
c-------------------------------------★接線剛性に関する増分ベクトル(vpp)
      call Add_tan_stiff_ld(ld_point_repeat,
     *       Member,n_member,est_ddisp_point,ak_nonlinear)
c          write(damp_out,*) ' Add_tan_stiff_ld_pa ok'
c-------------------------------------★Maxwell 型モデルの計算(vpp)
           call Add_fdd_ld(ld_point_repeat,E_model6_real,Element,
     *           Member,n_member,est_vel_point,rot_memb)
c          write(damp_out,*) ' Add_fdd_ld ok'
c-------------------------------------★右辺２項の和を取る(ok)
           call add_vec(n_unknown,ld_point_repeat,ld_point)
c          write(damp_out,*) ' add_vec ok'
c-------------------------------------★右辺項を送る (vpp)
c---------- ★★マスターに右辺項を送信
c          call reduce_ld_repeat(n_unknown,ld_point_repeat,
c     +          pa_work_ld,MPP_ANA_GROUP)
           call send_ld_repeat(n_unknown,ld_point_repeat)
c          write(*,*) ' send_ld_repeat ok'
c-------------------------------------★サーバーで計算した加速度を受け取る
 (vpp)
c---------- ★★マスターから計算結果加速度を取得
      call recv_result_acc(n_unknown,result_acc_point,ierr,
     *                     S_control_code)
c-------------------------------------★β法に基づき加速度より変位と速度を計
算(ok)
      call Cal_disp_vel(n_unknown,
     *          result_disp_point, result_vel_point, result_acc_point,
     *          past_disp_point, past_vel_point, past_acc_point,
     *          Newmark_P)
c          write(damp_out,*) ' Cal_disp_vel ok'
c---------- ★★収束したかチェック(mpp)    1:収束  0:未収束
c           スレーブ制御コード 0:  解析開始  正常終了の場合は最後は制御コードなし
c                              1:  発散・途中終了
c                              2:  エラー発生、途中終了
c                              3:  未収束
c 同期
c          call synchronous_process( MPP_Ana_Group)
c          call recv_ite(ite_end)
           if(S_control_code.eq.0)go to 9980                    ! 収束していればループから抜け
る
           if(S_control_code.eq.2)go to 9998
c          write(damp_out,*) ' Check_error ok'
c-------------------------------------★次の増分値を予測(ok)
           nx=iroop
           call Estimate_disp_vel(nx, n_unknown,
     *          est_disp_point, est_vel_point, est_ddisp_point,
     *          past_disp_point, past_vel_point, past_acc_point,
     *          result_disp_point, result_vel_point,
     *          past_dacc_point,result_acc_point, Newmark_P)
c          write(damp_out,*) ' Estimate_disp_vel ok'
      end do
c-----------------------------------------------------------------
c
c     ★          収束しなかった時の後処理
c
c-----------------------------------------------------------------
```

```
 9987 continue
      write(76,*) '  収束エラー発生、処理中止'
      goto 9997
c      return
c-----------------------------------------------------------------------
c
c     ★         計算終了・後処理開始
c
c-----------------------------------------------------------------------
 9980 continue
c-----------------------------------------------------★解析結果・増分変位をセット(ok)
      call Set_ddisp(n_unknown,est_ddisp_point,
     *     result_disp_point,past_disp_point)
c      write(damp_out,*) ' Set_disp_vel_acc ok'
C      print *, ' Set_disp_vel_acc ok'
c------- ★★  Parallel Code -Start (Stress)
c-----------------------------------------------------★部材応力を計算(vpp)
      call Cal_stress(Member,n_member,
     *     Model_type,Element,
     *     past_disp_point,past_vel_point,est_ddisp_point,rot_memb,
     *     E_model6_real,ak_nonlinear)
c-----------------------------------------------------------------------
c
c     ★         部材の弾塑性状態をチェック
c
c-----------------------------------------------------------------------
c-----------------------------------------------------★部材塑性をチェック
c-----------------------------------------------------★部材両端の節点力計算 (ok)
c-----------------------------------------------------★ファイバー応力セット(vpp)
      call Check_stress(Control,Control.type_analysis,
     *     ak_nonlinear,Member,n_member,Model_type,
     *     Element,past_disp_point,est_ddisp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,
     *     Bilinear_work,Trilinear_work,Concrete_work,RO_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member,
     *     S_comp_model,E_modelx,M_modelx,
     *     E_fiber_work,M_fiber_work)
C          write(damp_out,*) ' Check_stress ok',Newmark_P.max_repeat
c-----------------------------------------------------★部材の節点力と部材応力をマスターに送
る(vpp)
c-------- ★★マスターに部材応力を送信 (マスターがファイル出力する時のみ)
       if(i_print.eq.0) then
      call send_force_vpp(n_member,Member,pa_work_force)
      endif
c     call send_force(n_member1,n_member2,Member,pa_work_force)
c-----------------------------------------------★Maxwell 型モデルの非線形性チェック(ok)
      call Check_Maxwell_stress(Member,n_member,
     *     Element,E_model6_real,Newmark_P)
c     write(damp_out,*) ' Check_Maxwell_stress ok'
c-----------------------------------------------★応力等の最大値セット
      call Get_max_stress(Member,n_member,Max_stress)
c      write(damp_out,*) ' Get_max_stress ok'
c-----------------------------------------------★解析結果・変位、速度、加速度をセット(ok)
      call Set_disp_vel_acc(n_unknown,est_ddisp_point,
     *     result_disp_point, result_vel_point, result_acc_point,
     *     past_disp_point, past_vel_point,past_acc_point,
     *     past_dacc_point)                          ! past_dacc_point に過去の加速度をセットする
c      write(damp_out,*) ' Set_disp_vel_acc ok'
c-----------------------------------------------★終了か？ (ステップと最大値チェック)
      if(istep .ge. Newmark_P.nn_step  ) goto 9998
c-----------------------------------------------★接線剛性の計算(ok)
      call Get_nonlinear_stiff(Control.type_analysis,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,S_comp_model,E_modelx, M_modelx,
     *     E_fiber_work, M_fiber_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member )
c      write(damp_out,*) ' Get_nonlinear_stiff ok'
 9999 continue

c-----------------------------------------------------------------------
c
c     ★         時間増分更新
c
c-----------------------------------------------------------------------
c      iend_code = 0
c      ns_step=ns_step + n_step
```

```
c 同期
c     call synchronous_process( MPP_Ana_Group)
c     call COMMITQQ( damp_out)

c     return
c-----------------------------------------------------------------------
c
c     ★         応答解析終了処理
c
c-----------------------------------------------------------------------
 9998 continue
c-------- ★★  最大応力等を送信
      call send_max_stress(Parameter_C,Max_stress)
 9997 continue
c-----------------------------------------------------------------------
c
c     ★         動的配列の解放
c
c-----------------------------------------------------------------------
c----------------★         配列の動的領域を解放する（その１）  ok
      if(M_alloc(1).eq.1)then
      DEALLOCATE (Point,Element,Member,Max_stress)        !ok
c      DEALLOCATE (fll_static_point,am_point,fll_force_point)        !ok
      DEALLOCATE (am_member,rot_memb_t,rot_memb,ak_linear,ak_nonlinear)        !ok
      N=Parameter_C.n_local_coord
      if(N.ne.0) DEALLOCATE (rot_local)        !ok
      N = Parameter_C.n_S_comp_model
      if(N.ne.0) DEALLOCATE (S_comp_model)
      N = Parameter_C.nE_New_Element
      if(N.ne.0) DEALLOCATE (E_Fiber_work)
      endif
c----------------★         配列の動的領域を解放する（その２）ok
      if(   M_alloc(2).eq.1)then
      N=Model_type.n_m_damp
      if(N.ne.0) DEALLOCATE (ac_member)        !ok
      n= Model_type.n_e_New_fiber
      if(N.ne.0) DEALLOCATE (E_modelx)
      n= Model_type.n_m_New_fiber
      if(N.ne.0) DEALLOCATE (M_modelx)
      n=Parameter_C.nM_New_Element
      if(N.ne.0) DEALLOCATE (M_Fiber_work)
      endif
c----------------★         配列の動的領域を解放する（その４）
      if(   M_alloc(4).eq.1)then
c      DEALLOCATE (test_vector)                          !テスト用
      DEALLOCATE (disp_point,vel_point,acc_point,
     *     est_disp_point,est_vel_point,est_ddisp_point)
      DEALLOCATE (result_disp_point,result_vel_point,result_acc_point,
     *   past_disp_point,past_vel_point,past_acc_point,
     *   past_dacc_point)
      DEALLOCATE (ld_point,ld_point_repeat)
c      DEALLOCATE (a_vector,b_vector,fld_static)

      endif
c----------------★         配列の動的領域を解放する（その５）
      if( M_alloc(5).eq.1)then
      endif
cc      DEALLOCATE (work1_elememt,work2_elememt,work1_member,work2_member)
c----------------★         配列の動的領域を解放する（その２）
      if(   M_alloc(2).eq.1)then
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      DEALLOCATE (RO_work )
      endif
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      DEALLOCATE (MSS_work )             !ok
      endif
      n=Model_type.n_m_model(6)
      if(n.ne.0) then
      DEALLOCATE (E_model6_real )             !ok
      endif
      n= Model_type.n_e_model(11)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model11 )                 !ok
      endif
      n= Model_type.n_m_model(11)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model11 )                !ok
      endif
      n= Model_type.n_e_model(12)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model12 )                 !ok
      endif
      n= Model_type.n_m_model(12)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model12 )                !ok
      endif
      n= Model_type.n_m_model(18)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model13 )                !ok
      endif
      n= Model_type.n_e_model(18)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model13 )                 !ok
      endif
      n= Model_type.n_e_model(15)         !要素モデルの数
      if(n.ne.0) then
```

```
      DEALLOCATE (E_model15 )                !ok
      endif
      n= Model_type.n_m_model(15)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model15 )               !ok
      endif
      n= Model_type.n_e_model(13)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model21 )                !ok
      endif
      n= Model_type.n_m_model(13)         !部材モデルの数
      if(n.ne.0) then
       DEALLOCATE ( M_model21 )                !ok
      endif
      n= Model_type.n_e_model(14)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model22 )                !ok
      endif
      n= Model_type.n_m_model(14)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model22 )               !ok
      endif
      n= Model_type.n_e_model(16)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model31)               !ok
      endif
      n= Model_type.n_m_model(16)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model31 )               !ok
      endif
      n= Model_type.n_e_model(17)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model32)               !ok
      endif
      n= Model_type.n_m_model(17)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model32 )               !ok
      endif
      n= Model_type.n_e_model(19)         !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model33)               !ok
      endif
      n= Model_type.n_m_model(19)         !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model33 )               !ok
      endif

      endif
c―――――――――――★          配列の動的領域を解放する（その３）
      if( M_alloc(3).eq.1)then
      n= Model_type.nm_div_fmodel          !要素モデル内のサブ要素の数
      if(n.ne.0) then
      DEALLOCATE (M_model_fiber )                !ok
      endif
      n= Model_type.nm_div_felement         !ファイバー要素のエレメント最大数
      if(n.ne.0) then
      DEALLOCATE (E_model_fiber )                !ok
      endif
      endif
c―――――――――――★          配列の動的領域を解放する（その５）
      if( M_alloc(5).eq.1)then
      n= Model_type.n_m_bilinear          !  バイリニアの履歴要の数
      if(n.ne.0) then
      DEALLOCATE (Bilinear_work )                !ok
      endif
      n= Model_type.n_m_trilinear          !  トリリニアの履歴要の数
      if(n.ne.0) then
      DEALLOCATE (Trilinear_work )                !ok
      endif
      n= Model_type.n_m_Concrete          !  コンクリートの履歴要の数
      if(n.ne.0) then
      DEALLOCATE (Concrete_work )                !ok
      endif
      endif
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★          並列処理で使用した動的配列の解放
c
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――
      DEALLOCATE (pa_work_force)
c     DEALLOCATE (pa_work_stress)
c―――――――――――――――――――――――――――――――――――――――★並列処理通信グループの解法

c―――――――――――――――――――――――――――――――――――――――★計算終了コードセット
      iend_code =1
      close (damp_out)
      return
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★          解析終了
c
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――
      end
```